

**NASA
Technical
Paper
3207**

June 1992

Correlation and Prediction of Dynamic Human Isolated Joint Strength From Lean Body Mass

Abhilash K. Pandya,
Scott M. Hasson,
Ann M. Aldridge,
James C. Maida,
and Barbara J. Woolford

20101215144



1992

Correlation and Prediction of Dynamic Human Isolated Joint Strength From Lean Body Mass

Abhilash K. Pandya
*Lockheed Engineering & Sciences Company
Houston, Texas*

Scott M. Hasson
*Texas Women's University
Denton, Texas*

Ann M. Aldridge
*Lockheed Engineering & Sciences Company
Houston, Texas*

James C. Maida
and Barbara J. Woolford
*Lyndon B. Johnson Space Center
Houston, Texas*



National Aeronautics and
Space Administration
Office of Management
Scientific and Technical
Information Program

TABLE OF CONTENTS

	Page
ACKNOWLEDGMENTS	v
ABSTRACT	vii
1.0 INTRODUCTION	1
2.0 OBJECTIVE	3
3.0 METHOD	4
3.1 Data Collection	4
3.2 Data Reduction	18
3.3 Regression Model Development.....	22
4.0 RESULTS AND DISCUSSION.....	27
5.0 FUTURE DIRECTIONS.....	35
6.0 REFERENCES	36
7.0 APPENDICES	37
A. Polynomial Torque Coefficients	37
B. Lean Body Mass Coefficients	39
C. Population Coefficients.....	41
D. Torque Prediction Code	43

LIST OF TABLES AND FIGURES

	Page
Table 1 Lean Body Weight	18
Figure 1 LIDO multi-joint testing unit.....	5
Figure 2 Shoulder flexion and extension.....	6
Figure 3 Shoulder abduction and adduction.....	7
Figure 4 Shoulder internal and external	8
Figure 5 Elbow flexion and extension.....	9
Figure 6 Wrist flexion and extension.....	9
Figure 7 Closeup view of wrist flexion and extension.....	10
Figure 8 Wrist radial and ulnar deviation	11
Figure 9 Wrist pronation and supination	12
Figure 10 Hip flexion and extension.....	13
Figure 11 Hip abduction and adduction	13
Figure 12 Knee flexion and extension	14
Figure 13 Ankle plantarflexion and dorsiflexion.....	15
Figure 14 Closeup view of ankle plantarflexion and dorsiflexion.....	15
Figure 15 Front view of simulated ratchet pushing and pulling.....	17
Figure 16 Side view of simulated ratchet pushing and pulling	17
Figure 17 Data before and after visual editing	19
Figure 18 Example of the coefficient file format.....	21
Figure 19 Lean body mass versus mean torque.....	24-25
Figure 20 Polynomial regression fit to normalized data	26
Figure 21 Isolated joint curve: predicted versus measured	27
Figure 22 Shoulder joint absolute value error.....	29

LIST OF TABLES AND FIGURES

PAGE

Figure 23	Elbow joint absolute value error	30
Figure 24	Wrist joint absolute value error.....	31
Figure 25	Absolute value error for all motions.....	32
Figure 26	Lean body mass versus mean torque for ratcheting.....	34

ACKNOWLEDGMENTS

The investigators would like to thank the subjects for participating in this research. In addition, we would like to thank the following research assistants: R. Bluhm, M. Buzek, M. Mitchell, A. Perez and J. Ricafrente. Finally, we would like to acknowledge the late Linda S. Orr, whose contributions to research and practical work in computer simulation and graphics was invaluable to this project.

ABSTRACT

A relationship between a person's lean body mass and the amount of maximum torque that can be produced with each isolated joint of the upper extremity was investigated. The maximum dynamic isolated joint torque (upper extremity) on 14 subjects was collected using a dynamometer multi-joint testing unit. These data were reduced to a table of coefficients of second degree polynomials, computed using a least squares regression method. All the coefficients were then organized into look-up tables, a compact and convenient storage/retrieval mechanism for the data set. Data from each joint, direction and velocity, were normalized with respect to that joint's average and merged into files (one for each curve for a particular joint). Regression was performed on each one of these files to derive a table of normalized population curve coefficients for each joint axis direction and velocity. In addition, a regression table which included all upper extremity joints was built which related average torque to lean body mass for an individual. These two tables are the basis of the regression model which allows the prediction of dynamic isolated joint torques from an individual's lean body mass.

1.0 INTRODUCTION

A relationship exists between a person's lean body mass and the amount of maximum torque that can be produced with each isolated joint of the upper extremity. The use of an easily measured parameter (lean body mass) to predict dynamic isolated joint torque, which is time consuming to measure, would be extremely valuable. In this study, we have collected maximum dynamic isolated joint data on 14 subjects for the upper extremity and formulated a regression model which will allow prediction of a joint angle versus torque curve for a particular individual. There are three phases in our study: 1) data collection, 2) data reduction, and 3) model formulation.

Phase one of our study, data collection, involved measuring the maximum torque for all the upper extremity joints (shoulder, elbow, and wrist) at 4 velocities for 14 subjects. Data for three subjects' lower extremities (hip, knee, and ankle) were also collected. Torque was measured by using a LIDO multi-joint testing unit (Loredan Biomedical, Inc., West Sacramento, California). The subjects were positioned so that the axis of the joint was directly in line with the axis of the dynamometer goniometer. Dynamometer attachments were selected and used to isolate the joint being measured. In this manner, all joints were characterized for all axes of rotation over a range of velocities. The data were collected using the Loredan software, LIDO Active 3.3, on an IBM PC. For all cases, the data set consisted of torque and angle pairs. In addition, anthropometric data were also collected which included height, weight, age, sex, skin fold measures, and dimensional assessment according to the format specified in NASA Man-Systems Integration Standards (MSIS) document [1].

The second phase of the project, data reduction, began with transferring the data to a UNIX-based workstation (Silicon Graphics). These data were formatted into an ASCII file, noise filtered, reformatted, and reduced to a table of coefficients of second degree polynomials. The polynomial coefficients were computed using a least squares regression method. These polynomials represent the torque as a function of angle (i.e., $\text{torque} = a + b \cdot \text{angle} + c \cdot \text{angle}^2$, where a , b , and c are the polynomial coefficients). All the coefficients were then organized into look-up tables. These tables represent a compact and convenient storage/retrieval mechanism for our entire data set and are available upon request.

The third phase of our project involved model formulation. Data from each joint, direction and velocity, of an individual were normalized with respect to that joint's average and merged into files (one for each curve for a particular joint). Regression was performed on each one of these files to derive a table of normalized population curve coefficients for each joint axis, direction, and velocity. In addition, a regression table, which included all upper extremity joints, was built which related average torque to lean body mass for an individual. These two tables are the basis of the regression model which allows isolated joint curve prediction. Because of the limited number of subjects (3) for the lower extremity data set, no correlation to lean body mass is presented here for these measurements.

Finally, we have encapsulated the results of this study in a tool kit of software routines executing on a variety of platforms such as UNIX, DOS, and Macintosh machines (Appendix D). This code contains our isolated joint torque model (with all the tables of torque coefficients) and will allow the prediction of dynamic isolated joint torques from an individual's lean body mass.

2.0 OBJECTIVE

Prediction equations have been developed from lean body mass for isometric/static strength [2]. Few studies have utilized lean body mass to predict isokinetic or dynamic strength [3]. An absence of literature exists when correlating lean body mass to isolated joint isokinetic mean torque or torque over an entire range of joint motion. In addition, lean body mass has not been used to predict the mean torque for dynamic complex tasks involving multiple joints. The use of an easily measured parameter (lean body mass) to predict dynamic isolated joint torque would be of significant value.

The objective of this project is to develop prediction equations which can be used to calculate isolated joint torque (either mean torque or torque as a function of angle) and mean torque for a complex task from the measurement of a person's lean body mass.

Specific aims:

1. Document all data measurement and data processing techniques.
2. Develop prediction equations for:
 - a. Mean dynamic torque over an entire range of motion for a particular joint (shoulder, elbow, and wrist) and during a complex task (ratchet wrenching or an extravehicular task).
 - b. Dynamic torque-position curves over the entire joint range of motion from lean body mass for individual isolated joints (shoulder, elbow, and wrist).
3. Develop a set of software tools which simplify dynamic torque data access, manipulation, and prediction for the set of isolated joints measured.

3.0 METHOD

3.1 Data Collection

Subjects

Fourteen subjects (8 males and 6 females) aged 21 to 28 years volunteered and participated in this investigation. The study was evaluated and approved by the Institutional Review Board of the University of Texas Medical Branch at Galveston, Texas. All subjects were informed of potential risks and signed a consent to participate in the study.

Equipment

The LIDO multi-joint testing unit (Loredan Biomedical, Inc., West Sacramento, California, Figure 1) is an integrated system consisting of a dynamometer connected to a personal computer via RS232c lines. The force unit comes with a series of attachments and a subject bench which allows isolation of particular joints. The software allows precise control of the actuator head and the various modes of operation (e.g., isometric, isokinetic, and concentric). In addition, a database of subjects can be maintained and displayed with the provided graphical software. The unique feature of this system is that it outputs all data in a machine-readable form for more accurate data analyses. This system was used to measure all the isolated joint forces as well as the composite test cases.

Data Collection Procedures

Data collection for this project occurred over an 8-week period at the University of Texas Medical Branch, School of Allied Health Sciences, Human Performance Laboratory. The general procedure for evaluating all the upper and lower extremity joint movements and simulated ratchet maneuvers was the same. Specific subject and joint positioning for the isometric and isokinetic tests is described in each independent section. The time required to perform all of the dynamic and isometric shoulder, elbow, and wrist joint measurements was 1.5 to 2 hours; evaluation of the hip, knee, and ankle measures required 1.0 to 1.5 hours; and evaluation of the 3 ratchet wrench maneuvers required 0.5 to 1.0 hours per subject.

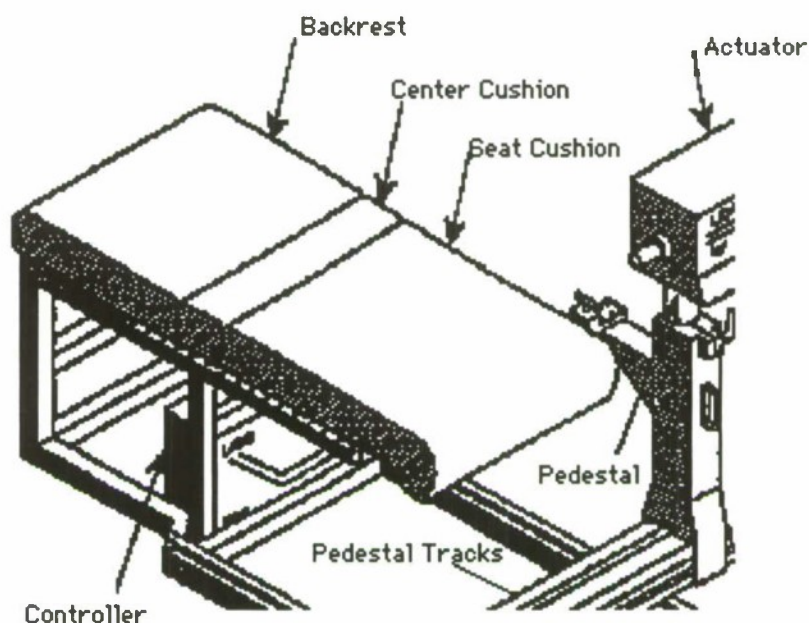


Figure 1. LIDO multi-joint testing unit.

On each testing day subjects reported to the laboratory in a fasting condition (food was restricted 3 hours prior to evaluation). For all upper extremity tests, the subject was positioned so that the axis of the joint was directly in line with the axis of the dynamometer goniometer. All measures were taken without gravity compensation. Maximum isometric contraction (MIC) measures were taken first. Dynamometer attachments were selected and placed to isolate the joint; the subject was positioned on the instrument and maximally stabilized; and then the joint was positioned at a specific angle. The subject was instructed each time to give maximum efforts. The subject performed three submaximal contractions followed by one MIC. The subject was given 3 minutes of rest and this procedure was repeated in the opposing direction. Next, the subject recovered for 5 minutes and then performed the isokinetic testing. The subject was instructed to give maximum efforts for each repetition and to move through the entire range of motion as rapidly and as forcefully as possible. The 4 joint velocity settings (60, 120, 180, and 240 deg/sec) were randomly assigned. The subject performed three submaximal contractions at the designated velocity followed by five maximum contractions. The subject was given 3 minutes of recovery between each velocity setting.

Shoulder Flexion and Extension

Subject was in a supinated position and was stabilized with velcro straps at the waist and across the clavicle just proximal to the shoulder to keep the right shoulder firmly in contact with the plinth (Figure 2). The angle for isometric testing was 90 degrees. A cuff attached to the dynamometer arm was placed on the upper arm just proximal to the elbow joint. Therefore, the shoulder joint was isolated and force was applied at the

point of the cuff attachment. The range of motion where shoulder flexion and extension torque was measured was between 20 and 180 degrees of shoulder flexion.

Shoulder Abduction and Adduction

Subject was lying on the side and was stabilized with velcro straps at the pelvis and across the upper chest at the axilla line to keep the subject's chest wall firmly in contact with the plinth (Figure 3). The angle for isometric testing was 90 degrees. A cuff attached to the dynamometer arm was placed on the upper arm just proximal to the elbow joint. Therefore, the shoulder joint was isolated and force was applied at the point of the cuff attachment. The range of motion where shoulder abduction and adduction torque was measured was between 15 and 145 degrees of shoulder abduction.

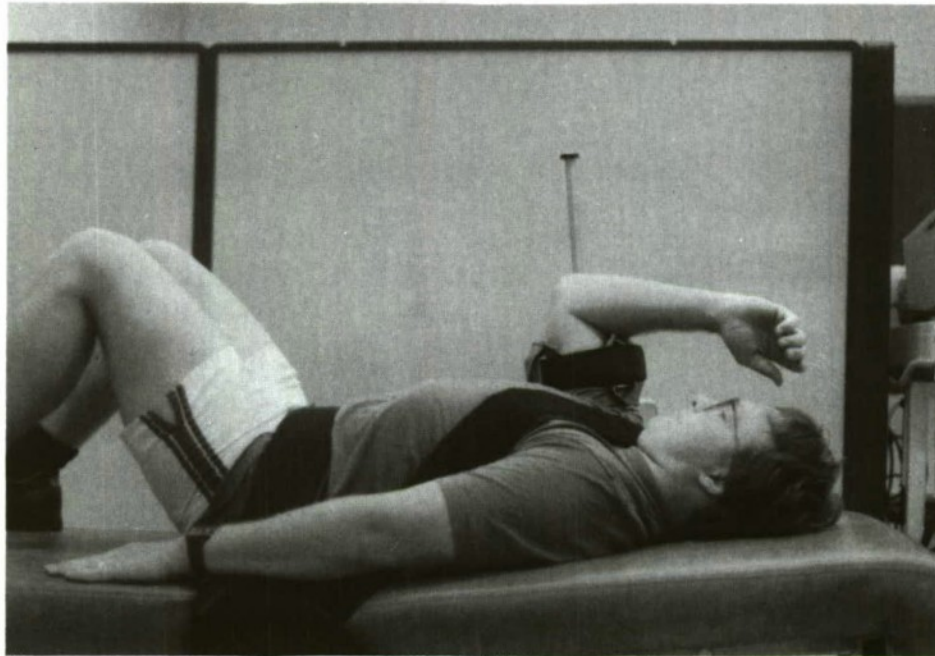


Figure 2. Shoulder flexion and extension.

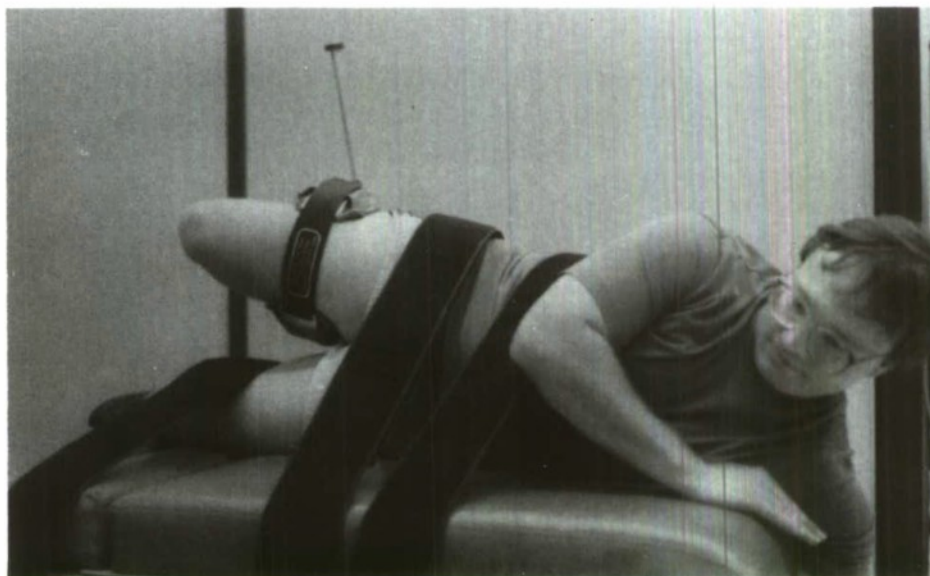


Figure 3. Shoulder abduction and adduction.

Shoulder Internal (Medial) and External (Lateral) Rotation

Subject was in a supinated position with the shoulder placed at 90 degrees of abduction. The subject was stabilized with velcro straps at the waist and across the mid-upper arm to keep the right-upper arm and elbow firmly in contact with the plinth (Figure 4). The angle for isometric testing was 0 degrees. A cuff attached to the dynamometer arm was placed on the forearm just distal to the elbow joint. The shoulder joint was isolated and force was applied at the point of the cuff attachment. The range of motion where shoulder internal and external rotation torque was measured was between 0 and 70 degrees of internal rotation and 0 and 60 degrees of external rotation.

Elbow Flexion and Extension

Subject was in a supinated position and was stabilized with velcro straps at the waist and across the chest and mid-upper arm to keep the right-upper arm and elbow firmly in contact with the plinth (Figure 5). The angle for isometric testing was 60 degrees. A cuff attached to the dynamometer arm was placed on the forearm just proximal to the wrist joint. The elbow joint was isolated and force was applied at the point of the cuff attachment. The range of motion where elbow flexion and extension torque was measured was between 15 and 135 degrees of elbow flexion.

Wrist Flexion and Extension

Subject was in a sitting position and was stabilized with velcro straps at the waist and across the chest to keep the subject's back firmly in contact with the seat (Figure 6). An attachment for the forearm was secured to the side of the LIDO. The subject's right forearm was secured in a supinated position into the device by three small velcro straps: 1) just distal to the elbow joint, 2) mid-forearm, and 3) just proximal to the wrist joint (Figure 7). This configuration was designed to keep the forearm firmly in contact with the forearm stabilizing device. The angle for isometric testing was 0 degrees. The subject gripped a handle device attached to the dynamometer shaft. The wrist joint was isolated and force was applied at the point of the handle attachment. The range of motion where wrist flexion and extension torque was measured was between 0 and 60 degrees for wrist flexion and 0 and 45 degrees for wrist extension.



Figure 4. Shoulder internal (medial) and external (lateral rotation).



Figure 5. Elbow flexion and extension.



Figure 6. Wrist flexion and extension.



Figure 7. Closeup view of wrist flexion and extension.

Wrist Radial and Ulnar Deviation

Subject was in a sitting position and was stabilized with velcro straps at the waist and across the chest to keep the subject's back firmly in contact with the seat (Figure 6). An attachment for the forearm was secured to the side of the LIDO. The subject's right forearm was secured in a supinated position into the device by three small velcro straps: 1) just distal to the elbow joint, 2) mid-forearm, and 3) just proximal to the wrist joint (Figure 8). This configuration was designed to keep the forearm firmly in contact with the forearm stabilizing device. The angle for isometric testing was 0 degrees (neutral). The subject gripped the handle device attached to the dynamometer shaft. Therefore, the wrist joint was isolated and force was applied at the point of the handle attachment. The range of motion where wrist radial and ulnar deviation torque was measured was between 0 and 35 degrees for wrist radial deviation and 0 and 35 degrees for wrist ulnar deviation.

Wrist (Forearm) Pronation and Supination

Subject was in a sitting position and was stabilized with velcro straps at the waist and across the chest to keep the subject's back firmly in contact with the seat (Figure 6). An attachment for the forearm was secured to the side of the LIDO. The subject's right forearm was in a neutral position and 2 small velcro straps were placed: 1) just distal to the elbow joint and 2) mid-forearm (Figure 9). This configuration was designed to keep the proximal section of the forearm firmly in contact with the forearm stabilizing device and to allow pronation and supination to occur. The angle for isometric testing was 0 degrees (neutral). The subject gripped the handle device attached to the

dynamometer shaft. Therefore, the forearm was isolated and force was applied at the point of the handle attachment. The range of motion where wrist pronation and supination torque was measured was between 0 and 60 degrees for wrist pronation and from 0 to 60 degrees for wrist supination.



Figure 8. Wrist radial and ulnar deviation.



Figure 9. Wrist (forearm) pronation and supination.

Hip Flexion and Extension

Subject was in a supinated position and was stabilized with velcro straps at the chest and waist and across the left-upper thigh to keep the back and pelvis firmly in contact with the plinth (Figure 10). The angle for isometric testing was 90 degrees. A large cuff attached to the dynamometer arm was placed on the thigh just proximal to the knee joint. Therefore, the hip joint was isolated and force was applied at the point of the cuff attachment. The range of motion where hip flexion and extension torque was measured was between 0 and 110 degrees of hip flexion.

Hip Abduction and Adduction

Subject was lying on the side and was stabilized with velcro straps at the chest and waist and across the left-upper thigh to keep the subject's chest wall and pelvis firmly in contact with the plinth (Figure 11). The angle for isometric testing was 0 degrees. A large cuff attached to the dynamometer arm was placed on the thigh just proximal to the knee joint. Therefore, the hip joint was isolated and force was applied at the point of the cuff attachment. The range of motion where hip abduction and adduction torque was measured was between 0 and 60 degrees of hip abduction.

Hip Internal (Medial) and External (Lateral) Rotation

Subject sat on an elevated table and was stabilized with velcro straps at the waist and across the left-upper thigh to keep the subject's pelvis firmly in contact with the exterior table. The angle for isometric testing was 0 degrees (neutral). A cuff attached

to the dynamometer arm was placed on the lower leg just distal to the knee joint. Therefore, the hip joint was isolated and force was applied at the point of the cuff attachment. The range of motion where hip internal and external rotation torque was measured was between 0 and 15 degrees of internal rotation and from 0 to 35 degrees of external rotation.



Figure 10. Hip flexion and extension.



Figure 11. Hip abduction and adduction.

Knee Flexion and Extension

Subject was in a sitting position and was stabilized with velcro straps at the waist and across the chest to keep the subject's back firmly in contact with the seat (Figure 12). The subject's right thigh was stabilized with an attachment secured to the side of the LIDO. This configuration was designed to keep the thigh firmly secured between the stabilizing device and the seat. The angle for isometric testing was 60 degrees. A cuff attached to the dynamometer arm was placed on the lower leg just proximal to the ankle joint. Therefore, the knee joint was isolated and force was applied at the point of the cuff attachment. The range of motion where knee flexion and extension torque was measured was between 5 and 100 degrees of knee flexion.

Ankle Plantarflexion and Dorsiflexion

Subject was in a supinated position with the knee joint fully extended. The subject was stabilized with velcro straps at the chest and waist to keep the subject's back and pelvis firmly in contact with the plinth (Figure 13). The subject's right knee was stabilized with an attachment secured to the side of the LIDO. This configuration was designed to keep the knee joint firmly secured between the stabilizing device and the plinth. The angle for isometric testing was 0 degrees (neutral). The subject's right foot was secured by three small velcro straps and placed into a foot plate device (Figure 14). The foot plate device was attached to the dynamometer shaft. Therefore, the ankle joint was isolated and force was applied at the foot plate attachment. The range of motion where ankle plantarflexion and dorsiflexion torque was measured was between 0 and 30 degrees for ankle plantarflexion and from 0 to 10 degrees for ankle dorsiflexion.



Figure 12. Knee flexion and extension.



Figure 13. Ankle plantarflexion and dorsiflexion.



Figure 14. Closeup view of ankle plantarflexion and dorsiflexion.

Simulated Ratchet Pushing and Pulling Maneuver Torque

The axis of the dynamometer goniometer was positioned in line with the subject's greater trochanter. For all simulated ratchet pushing and pulling maneuvers, the subject was in a sitting position. The right upper extremity was evaluated. The subject was stabilized with velcro straps at the waist and across the chest to keep the subject's back firmly in contact with the seat (Figure 15). The subject gripped a simulated ratchet device at a height of 90% of the linear distance measured from the subject's greater trochanter to the acromioclavicular joint (Figure 16). The subject was instructed to move the ratchet device from a full back position (shoulder joint at 10.9 ± 0.8 degrees; elbow joint at 112.0 ± 6.0 degrees) to a full forward position (shoulder joint at 66.0 ± 6.8 degrees; elbow joint at 0 degrees) [ratchet push maneuver], and then to return to the starting full back position [ratchet pull maneuver]. The subject was told to give maximum efforts for each repetition and to move through the entire range as rapidly and forcefully as possible, but to not allow the scapula (upper back) to lose contact with the seat. Individual subject measures of ratchet grip height and shoulder and elbow joint angle of excursion were recorded. For all ratcheting conditions, the subject performed three submaximal contractions followed by five maximum contractions. The subject was given 5 minutes of recovery between each of the 3 ratcheting conditions. The first ratcheting condition consisted of two trials. The first trial consisted of a constant resistance of 25 ft-lbs (isotonic load) for the ratchet push maneuver and 0 ft-lbs of resistance for the pull maneuver. The subject was given a 5-minute recovery period and then performed the second trial with 0 ft-lbs of resistance for the ratchet push maneuver and a constant resistance of 25 ft-lbs for the pull maneuver. The second condition was the push and pull ratcheting done at a constant angular velocity of 120 deg/sec (isokinetic). The third condition was the push and pull ratcheting done at a constant angular velocity of 240 deg/sec (isokinetic). For each repetition, torque and angle position data were measured.

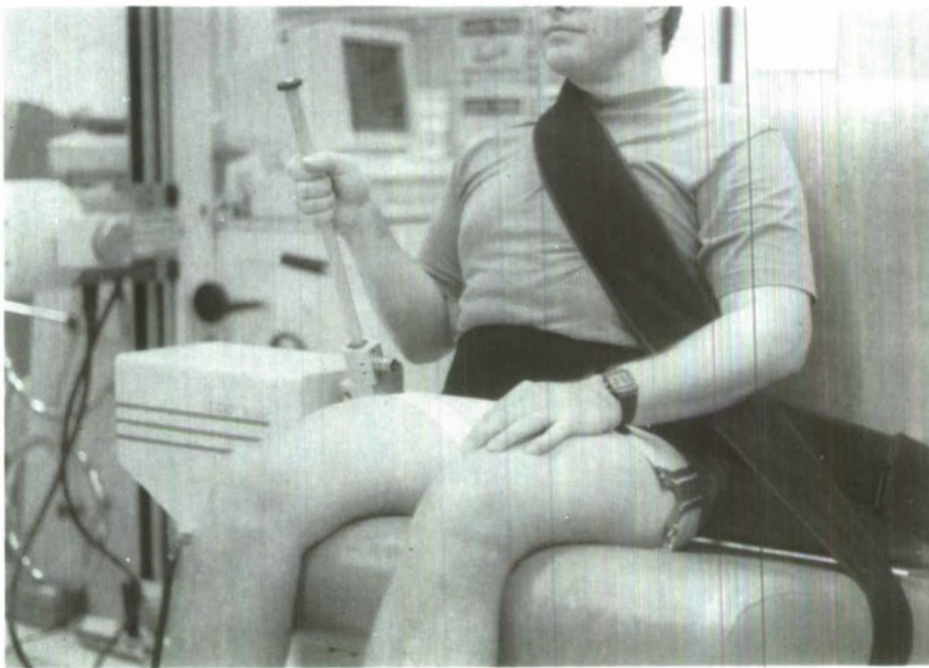


Figure 15. Front view of simulated ratchet pushing and pulling maneuver.



Figure 16. Side view of simulated ratchet pushing and pulling maneuver.

Assessment of Anthropometric Data

Anthropometric data were gathered for height, weight, age, sex, skin fold measures, and dimensional assessment. Height was measured in centimeters and weight in kilograms on a physician scale. Skin fold measures were taken with a Lange skin fold caliper. Males were measured at the abdomen and anterior thigh and chest; and females were measured at the anterior thigh, suprailiac crest, and triceps. Estimation of body fat and lean body mass were performed using equations specific for sex and age [4]. See Table 1 for a summary of this data.

TABLE 1. Lean Body Weight

SUBJ	SEX	AGE	HT (cm)	WT (kg)	%Body Fat	Lean Body
1.	M	23	173	64.9	10.4	58.2
2.	M	25	178	76.0	10.4	68.1
3.	M	28	188.3	84.5	14.5	72.2
4.	F	23	172.3	7.2	29.5	54.4
5.	M	22	185.0	88.2	10.7	78.8
6.	M	25	180.0	86.0	5.7	81.1
7.	M	26	176.3	95.2	19.0	77.1
8.	F	23	174.0	60.6	18.6	49.3
9.	F	22	168.0	59.1	14.8	50.4
10.	F	21	158.5	46.4	17.2	38.4
11.	F	21	158.0	51.6	19.5	41.5
12.	M	21	178.5	80.7	8.9	73.5
13.	F	23	166.0	55.7	20.8	44.1
14.	M	23	162.5	68.5	11.3	60.8

The dimensional anthropometric data were taken with cloth tape measures [1]. The time required to make all anthropometric measures was 0.25 to 0.5 hours per subject.

3.2 Data Reduction

A set of streamlined programs was developed to process the raw strength data (produced directly by the LIDO force torque dynamometer) into a compact polynomial coefficient format. The raw data were collected using the LIDOACT software executing on an IBM PC. The files produced on the PC were transferred to the VAX system using a data communication software package (Kermit) in binary mode. These data were then transferred to a UNIX-based workstation.

After being separated into files by subject, velocity, direction, and degree of freedom for each joint, each torque versus angle data file was viewed graphically and edited for extraneous data points (Figure 17). Figure 17 shows that the initial and final portions of the curves were omitted because of the startup time during which the subject is beginning to apply a maximum torque. At the end of a motion, the subject was anticipating the stopping and change of direction of the LIDO actuator arm. These

transition regions of torque were inconsistent and so were not part of our modeling effort.

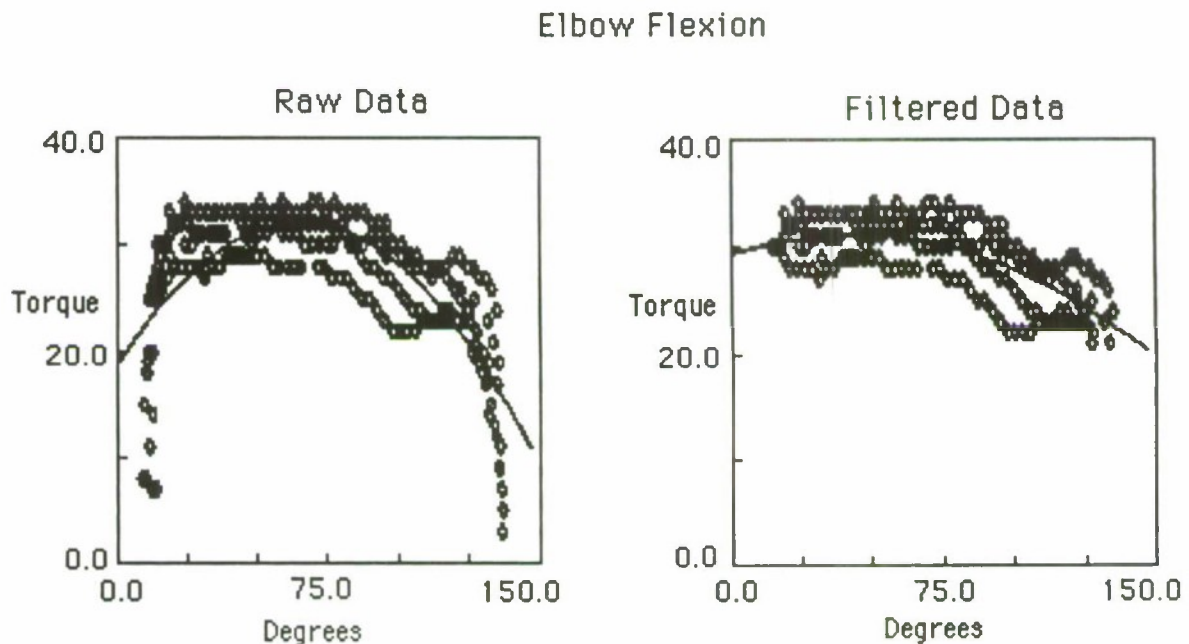


Figure 17. Data before and after visual editing.

The following is the flow of data from raw files to a torque function coefficient file (Figure 18). This is the actual UNIX script file used:

```
#uncompress the large data files
uncompress *.raw
"ls" sh*.raw| lido | tosort | sort | toffc > right_shoulder.ffc
"ls" el*.raw| lido | tosort | sort | toffc > right_elbow.ffc
"ls" wr*.raw| lido | tosort | sort | toffc > right_wrist.ffc
#clean up
compress *.raw
mv *.xy* xy
rm *.asc
```

The following is an explanation of the above script file:

1. The first line of the script file uncompresses the data files needed.
2. The "ls" command feeds a list of the file names (one at a time) to the LIDO program.

3. The LIDO program converts each file it receives from the LIDO format into an ASCII format and passes the data (one file at a time) to tosort.
4. The tosort program computes the regression equations and collates person data and passes that condensed data on to the sort program. In addition, it creates xy files of force versus angle and stores them for review later.
5. The sort program sorts the data on each field and passes that on to the toffc program.
6. The toffc program processes sorted data to produce files in the torque function coefficient format (Figure 18).
7. The original data is then recompressed in the seventh line to conserve disk space.
8. The xy files are moved into a separate directory called xy in the eighth line.
9. All the unnecessary files are cleaned up in the ninth line.

The following is a quick description of each of the processing programs:

lido.c - This program converted a raw LIDO generated data file to a standard output ASCII file. The LIDO generated data file (extension .raw) had a main header area and several subheader areas. Lines in these areas were terminated with a line feed as in a normal ASCII text file. The data lines were terminated with a carriage return and no line feed. The data lines were three integer values of four characters each. The first value was the corrected torque at the cuff [5, 6], the second value was the angle, and the third was the uncorrected torque at the shaft of the LIDO actuator. The corrected torque values were chosen for further processing.

tosort.c - This program converted the ASCII LIDO data file to torque function coefficients and an associated subject name, joint name, direction, and velocity. The output was used as standard input for the UNIX sort utility. This sorted output was then piped to the toffc.c program.

toffc.c - This was a program which processed output from the tosort program and converted it into a torque coefficient file (Figure 18).

```

right_shoulder /* joint name*/
x              /* axis */
abduction      /* direction*/
4              /*number of velocities*/

/* 1 velocity , 3 polynomial coefficients. note: Y = A + Bx + Cx**2 where A,B,C are the coefficients*/

60.000000      3.477892E+01      2.220639E-01      -2.821324E-03
120.000000     5.058879E+01      -1.974063E-01     -2.446309E-04
180.000000     3.441185E+01      -2.165272E-02     -1.170670E-03
240.000000     4.630580E+01      -2.306546E-01      6.999267E-05
adduction
4
60.000000      3.899835E+01      -2.004480E-02     -1.001798E-04
120.000000     3.210880E+01      6.244854E-02      -3.269533E-04
180.000000     1.128712E+01      4.013751E-01      -1.763708E-03
240.000000     2.879990E+01      -4.255312E-02      6.872315E-04
y
extension
4
60.000000      1.853307E+01      1.233500E-01      -1.150867E-04
120.000000     1.598296E+01      2.713108E-01      -9.975418E-04
180.000000     1.516010E+01      2.109192E-01      -8.056303E-04
240.000000     9.103311E+00      1.325615E-01      -4.602912E-05
flexion
4
60.000000      6.689232E+01      -4.941486E-01      1.258664E-03
120.000000     4.769003E+01      -2.906334E-01      8.198689E-04
180.000000     5.670030E+01      -4.014365E-01      1.149240E-03
240.000000     4.729467E+01      -3.183639E-01      8.940246E-04
z
lateral
4
60.000000      1.650289E+01      -4.517265E-02      -5.588200E-04
120.000000     1.434598E+01      -2.340409E-01      -2.806892E-03
180.000000     1.873110E+01      -1.003202E-01      -1.389751E-03
240.000000     1.489092E+01      -4.907729E-02      4.311464E-04
medial
4
60.000000      2.395702E+01      1.128039E-01      -1.947907E-04
120.000000     2.124104E+01      1.564447E-01      -3.186225E-03
180.000000     2.144591E+01      1.826617E-01      -1.797236E-03
240.000000     1.980874E+01      1.336217E-01      -8.705539E-04

```

Figure 18. Example of the coefficient file format.

3.3 Regression Model Development

Creating the regression model involved processing the data from each joint, direction and velocity, into separate files. There were three phases in the processing of these files which allow the prediction of individual torque curves.

1. Creating lean body mass to average torque relationship tables.
2. Generating a population shape coefficient table.
3. Linking the population shape table to the lean body mass average torque table.

Lean Body Mass to Average Torque Table Generation

A relationship exists between lean body mass and average torque. Figure 19 shows representative examples of this linear relationship for the shoulder, wrist, and elbow. Correlation coefficient values range from 0.80 to 0.95. The lean body mass for each of the subjects was calculated from the measured weight and % body fat:

$$\text{lean body mass} = \text{weight} * (1 - \% \text{ body fat})$$

The average torque was then computed for the entire joint range for each velocity of each joint. From these data, linear regression coefficients were obtained and organized into coefficient tables. These tables can be used to calculate the average torque for a particular joint and velocity when the lean body mass is known.

Obtaining Population Curves

In our model an assumption was made that for an isolated joint motion, individuals use the same muscle groups. These muscle groups basically have the same shape, orientation, and points of attachment and differ in the magnitude of force exerted. Hence, when the torque-position curves are normalized and plotted, a general trend can be seen for a particular joint. Figure 20 represents the normalized data for all the subjects for the elbow and shoulder flexion. These data show a definite trend. To take advantage of this relationship, the data were processed in the following way:

1. We normalized each individual data set (isolated joint for each axis, direction, and velocity) with respect to the average torque for that data set.
2. For all subjects, the normalized data for an isolated joint (axis, direction, and velocity) were combined into one file. A total of 7 axes, 2 directions, and 4 velocities resulted in 56 composite files of normalized subject data. (Figure 20 shows a sample case.)
3. A regression for each of these combined files was then computed.
4. The second order polynomials, representing the shapes of the joint motions, were then organized into coefficient tables (Figure 18). These tables allow

the computation of a normalized torque-position curve for any joint, axis, direction, and velocity.

Generation of an Individual's Torque-Position Curve

To generate a torque-position curve from lean body mass information it is necessary to multiply the normalized torque-position curve coefficients (established in the table from step 2) by the average torque. The average torque is computed from the data in the lean-body-mass to average-torque-coefficient table established in step 1 above. These two tables provide the data to compute a polynomial which represents an individual's torque-position curve for that joint, axis, direction, and velocity.

For all of our data, torque was measured for four separate velocities. No attempt was made to find a correlation between these velocities. Velocity dependence was not consistent among all joints. Since dynamic strength is important in most joint motions, we feel that this area requires more attention.

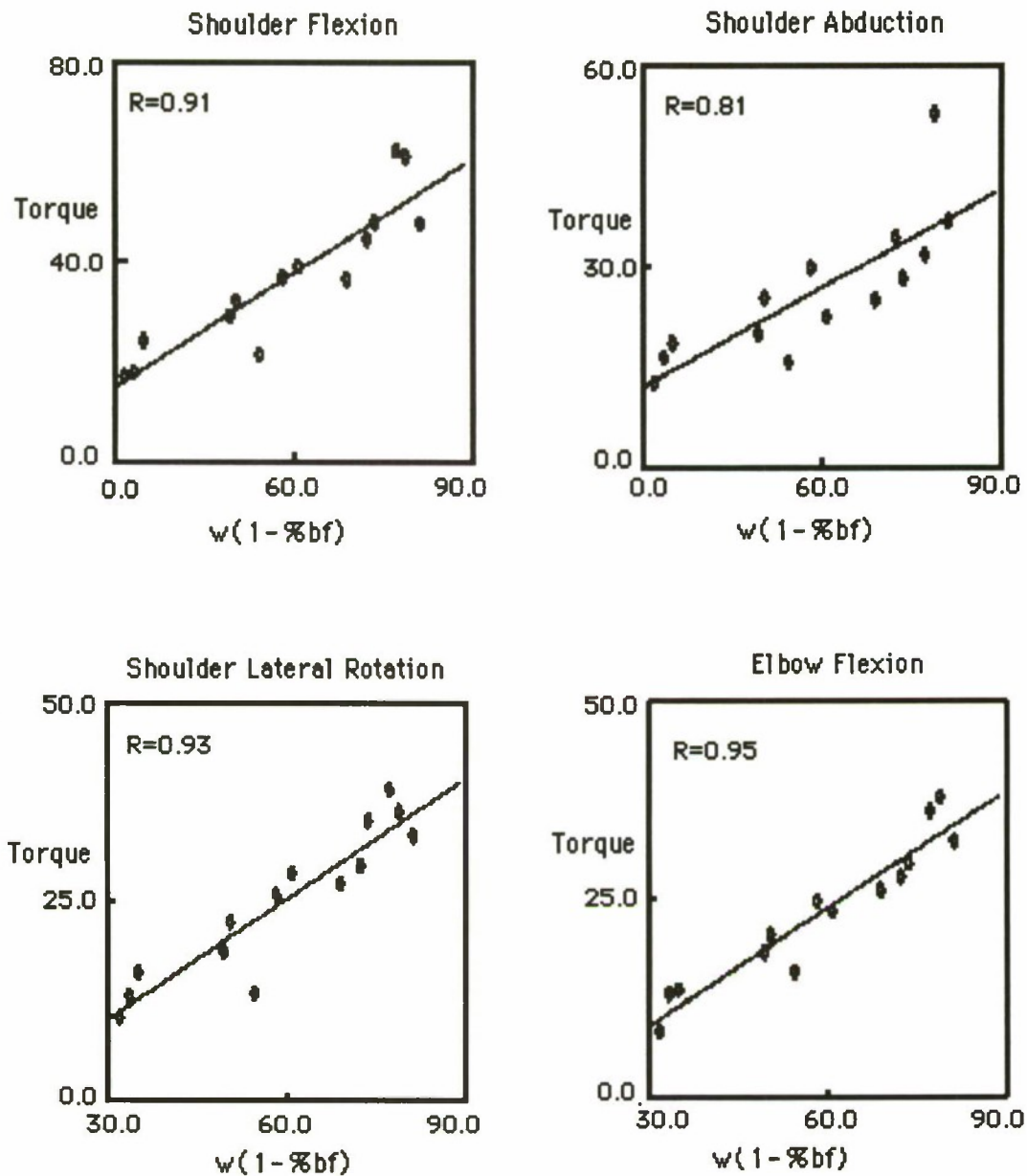


Figure 19. Lean body mass (kg) versus mean torque (ft-lbs).

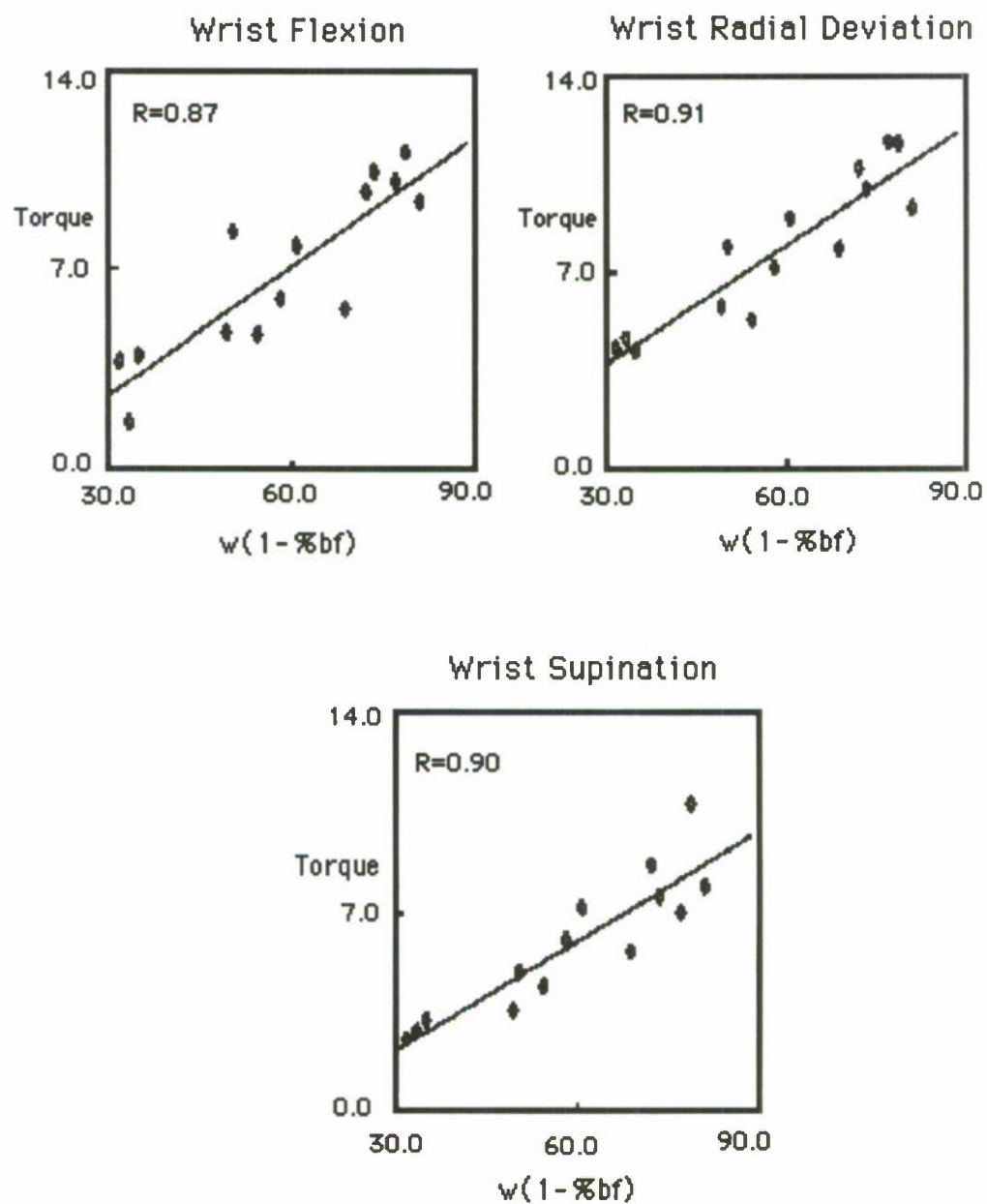
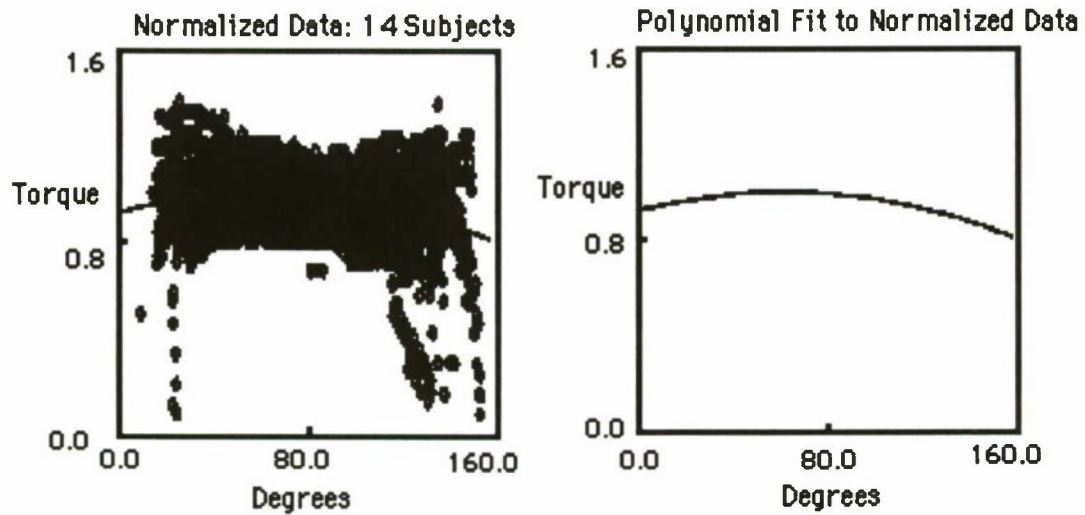


Figure 19. Lean body mass (kg) versus mean torque (ft-lbs) (continued).

Elbow Flexion: velocity = 120 deg/sec



Shoulder Flexion: velocity = 120 deg/sec

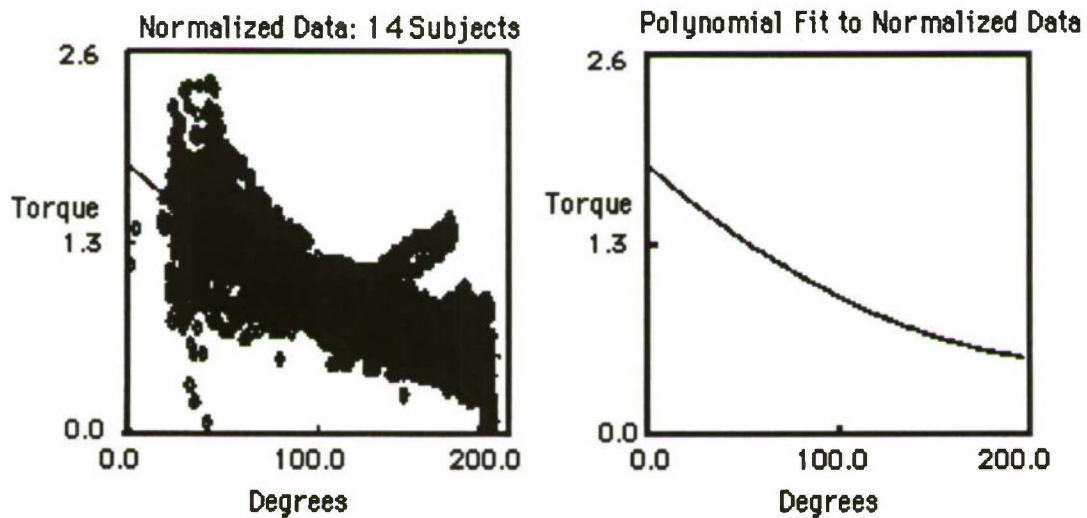


Figure 20. Polynomial regression fit to normalized data.

4.0 RESULTS AND DISCUSSION

Fourteen subjects were measured for isolated joint torques in each axis, direction, and for four different velocities. This study shows how these time-consuming measurements of dynamic isolated joint torques can be predicted. A strong correlation between torque produced by an isolated joint and the individual's lean body mass was found. This relationship allows prediction of the torque-position curves for each isolated joint by a simple index, the lean body mass.

To verify our regression model for predicting isolated joint torque-position curves, we plotted predicted versus measured torque-position curves. Figure 21 is an example of a prediction of an isolated joint torque-position curve (from lean body mass information only) plotted with measured data. The general shape of the curves of the data are reproduced.

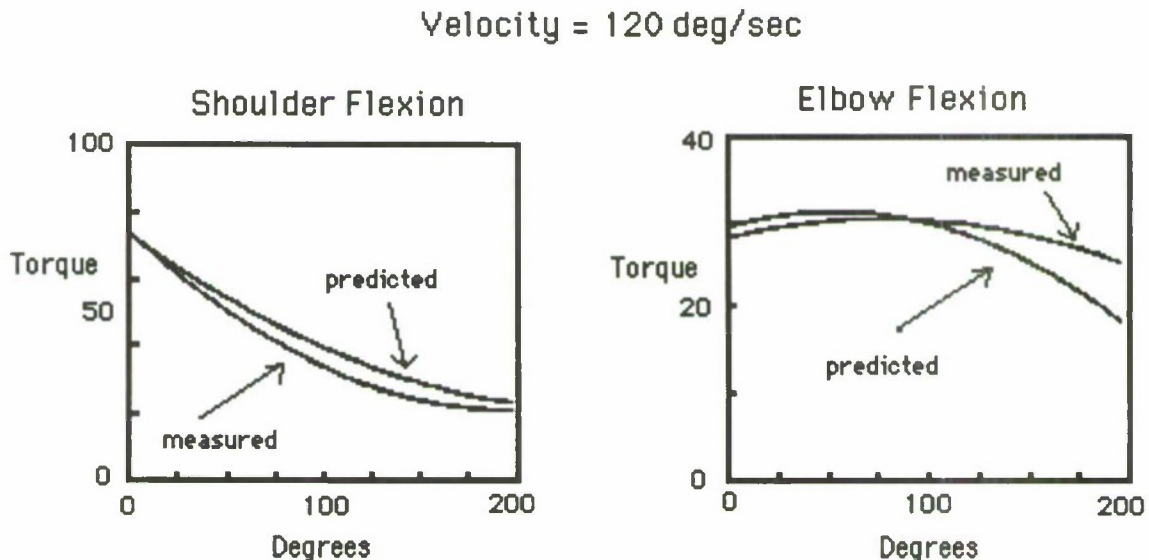


Figure 21. Isolated joint curve: predicted versus measured.

To get an estimate of the deviation from the measured data, a comparison computation was done. For each of the torque-position data sets collected, a corresponding array of torque values was calculated from the predicted polynomial coefficients for that particular data set. A torque difference vector (a difference of the actual measured torque value at that angle and the computed value at that point from the predicted polynomial coefficients) was then created. A percentage absolute value (relative to the maximum) of the difference array was calculated and plotted. This same analysis was done on the regression coefficients computed from that particular data set. The regression coefficients represent the ideal curve through the collected data values.

Figures 22, 23, and 24 represent a comparison of the error of the measured data fit regression coefficients versus the error from the lean body mass predicted coefficients for one representative subject. The error of the regression coefficients reflects the

deviation in the fit to the measured data. In these figures, the average error of the fit to the measured data is 6%. For prediction of second order regression coefficients from lean body mass information, this is the best that one can hope to achieve. As expected, the predicted coefficients from lean body mass have a greater deviation than the regression coefficients calculated from the actual data. For the shoulder flexion/extension, elbow flexion/extension, and wrist flexion/extension the deviations are 9.8% for this individual. Figure 25 represents the same comparison for all joints for all individuals. When compared across the entire subject pool, the shoulder and elbow joint prediction equation represents about a 12% absolute value deviation. The wrist joint has a greater deviation. The wrist data have, on the average, one-tenth the range of the shoulder and elbow joints. As a result of this small range, a comparatively small deviation translates into a large percentage difference. The wrist data need further consideration.

If only lean body mass data on a particular individual are available, our results indicate that the prediction equations can yield results on the order of 12% deviation from the actual data for all axes and directions of the elbow and shoulder joints. The results here could also be used to fill in missing or partial strength information for an individual. For instance, if only one axis of a particular joint was measured, the other strength information of the other axes and directions could be extrapolated from our equations taking into account the known information. Other correlates to dynamic strength prediction may exist. An area of future research is finding multiple correlates of strength.

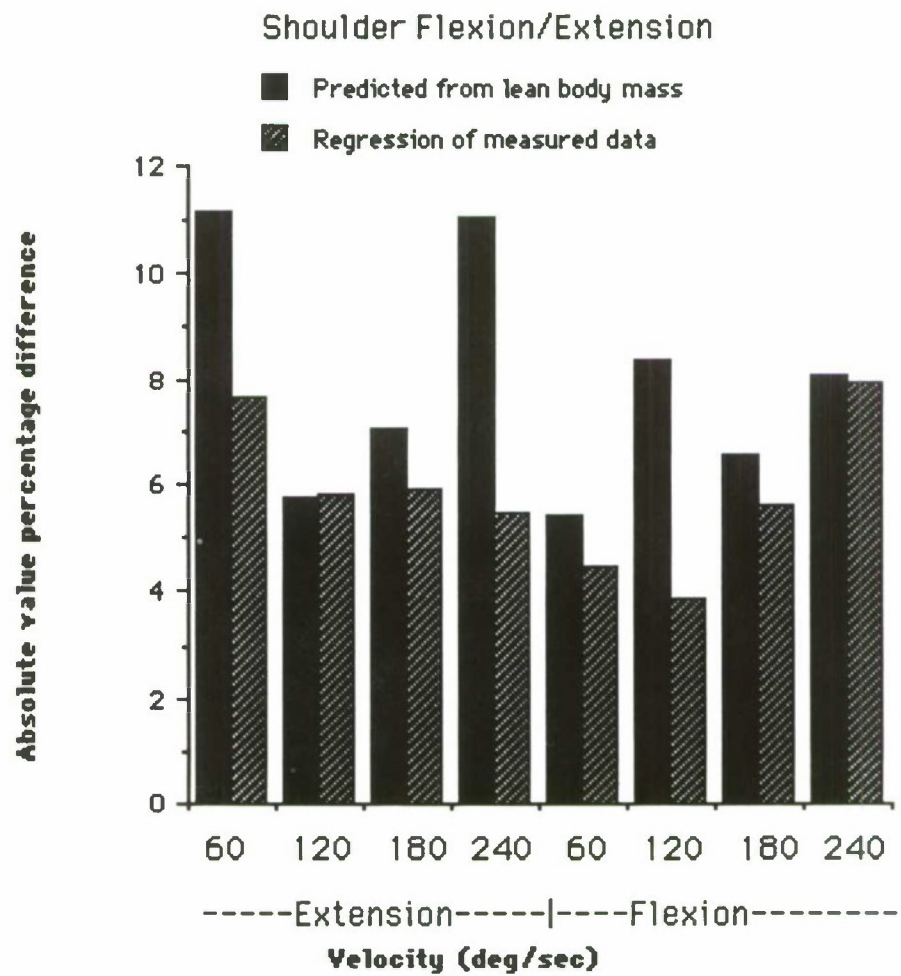


Figure 22. One subject's shoulder joint absolute value error of the predicted curve as compared to the error of the regression curve generated through the collected data.

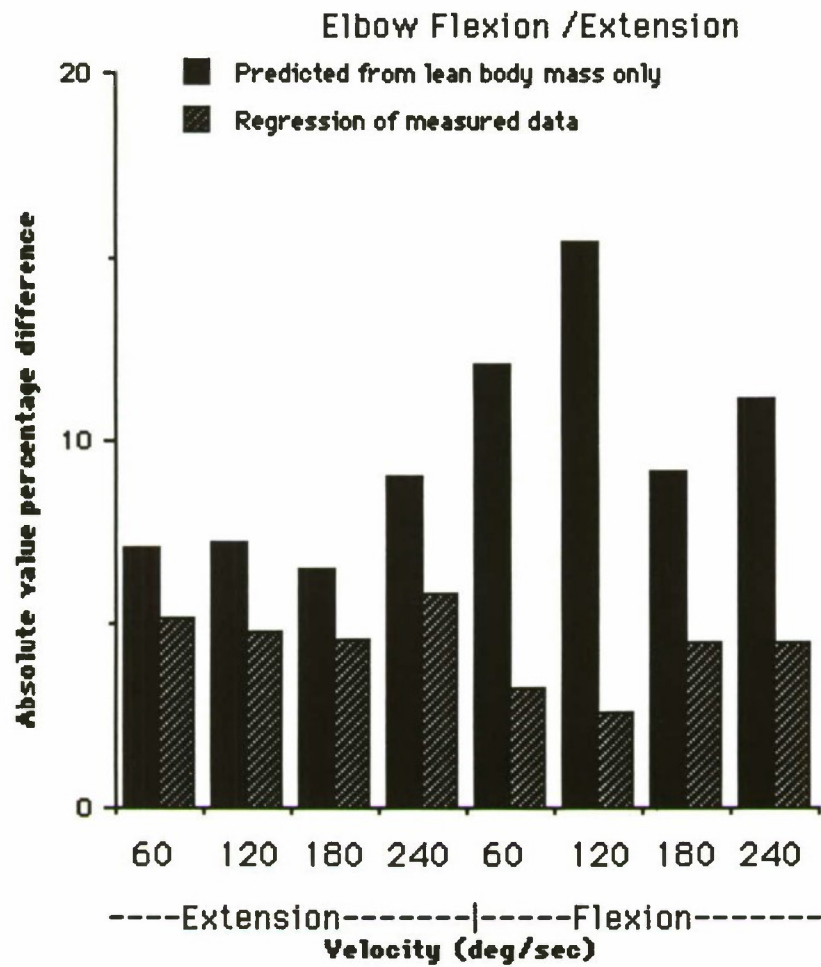


Figure 23. One subject's elbow joint absolute value error of the predicted curve as compared to the error (variation) of the regression curve generated through the collected data.

Wrist Flexion/Extension

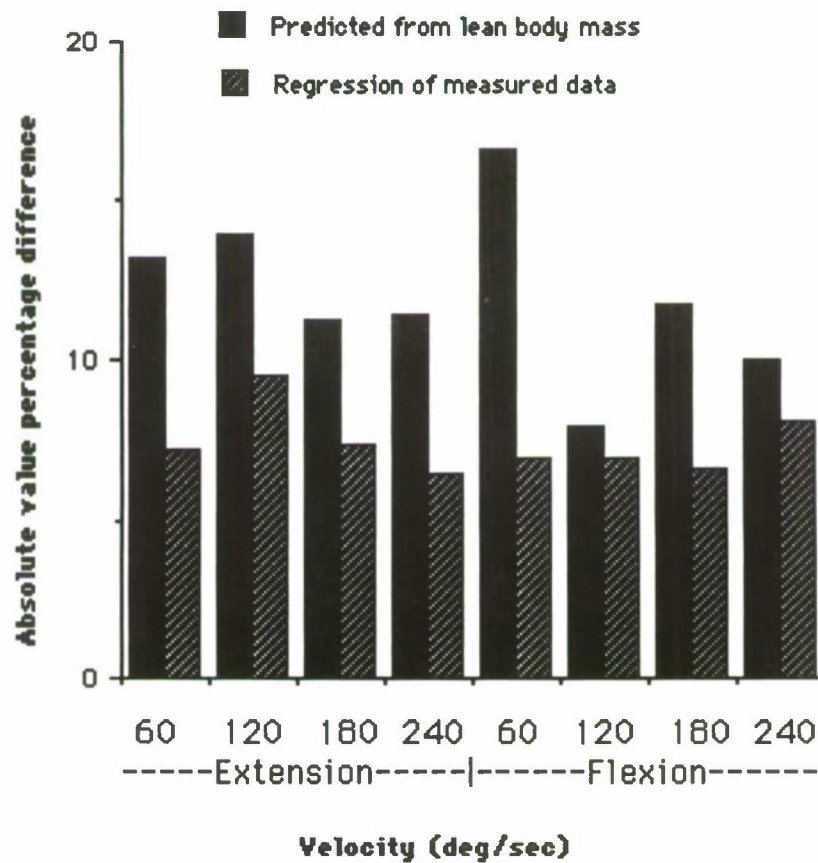


Figure 24. One subject's wrist joint absolute value error of the predicted curve as compared to the error (variation) of the regression curve generated through the collected data.

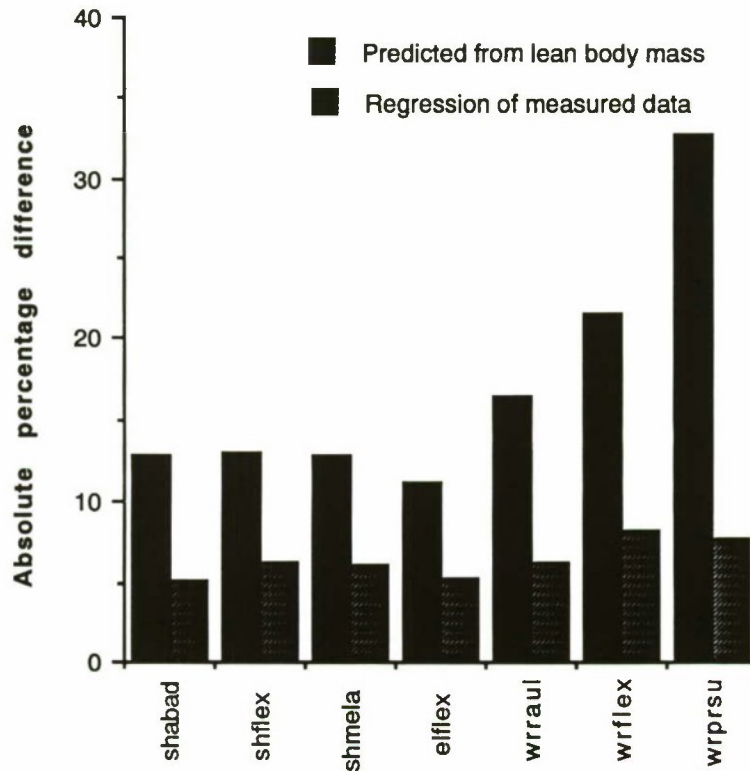


Figure 25. Entire subject pool. This is the absolute value error of the predicted curve as compared to the error (variation) of the regression curve for all subjects for all axes of the joints.

Our model was based on measurements of 14 subjects. These subjects were all healthy young adults. Extending our lean body mass-torque regression model beyond this scope would require additional validation.

Our results include several useful tables:

1. Tables of second order polynomial coefficients were generated from the measurements for each joint, axis, direction, and velocity. These tables provide a convenient and compact storage and retrieval mechanism to access our entire measured subject pool data of dynamic isolated joint torques (Appendix A).
2. Tables of first order polynomial coefficients relating lean body mass of an individual to the average torque for an isolated joint motion were generated. They allow the calculation of an average torque for a particular joint, direction, and velocity from an individual's lean body mass (Appendix B).

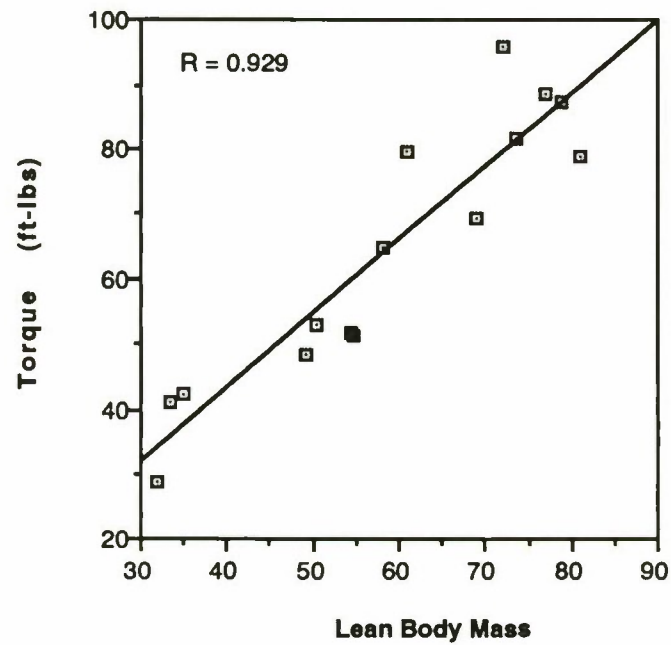
3. Also available are second order coefficient tables characterizing the torque-position curves that have been normalized across our subject pool. These curves represent the general (normalized) trends in a torque-position curve and may be useful in studying torque as a function of joint angle in a population (Appendix C).

To make efficient and practical use of all the coefficients, variables, and relationships found in this report, we have encapsulated the results of this study in a tool kit of software routines executing on a variety of platforms such as UNIX, DOS, and Macintosh machines (Appendix D). This code contains our isolated joint torque model (with tables of torque coefficients) that allows the prediction of dynamic isolated joint torques from an individual's lean body mass.

Although our report has focused on isolated joint prediction, we did measure and correlate a multi-joint task (the ratchet wrench push-pull) to lean body mass. Figure 26 indicates that multi-joint tasks may also be related to lean body mass. Percentage body fat is a good predictor of torque. There is a strong correlation ($r > 0.92$) between torque production capability and the lean body mass. Once a representative sample of a population has been measured for a particular composite motion, joint strength and prediction of torque capability of a particular individual may be extrapolated by only two measures: percentage body fat and weight. Research continues in this area.

In addition, this ratchet data as well as all the isolated joint data are the basis of a graphically based human strength model being developed at the NASA-JSC Man-Systems Division. This model calculates end effector strength based on any arbitrary motions using this dynamic isolated joint information and a vector sum algorithm.

Lean body mass vs. torque for ratcheting task (push)



Lean body mass vs. torque for ratcheting task (pull)

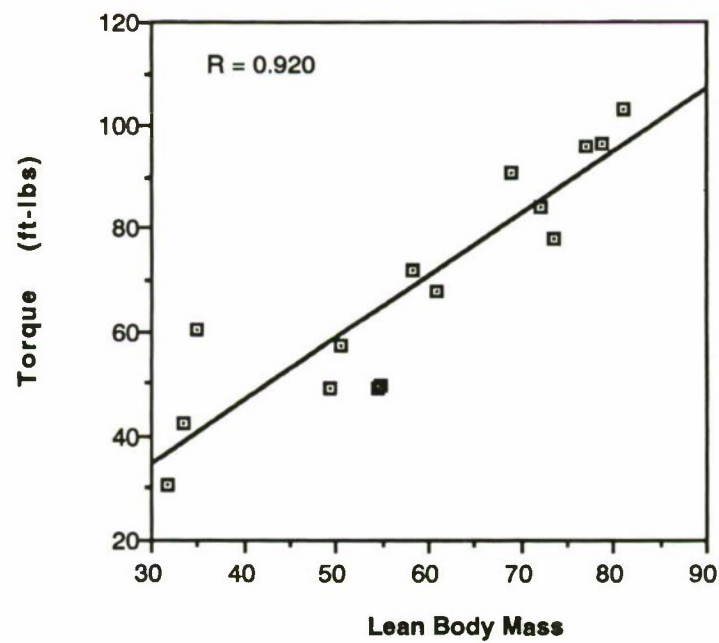


Figure 26. Lean body mass versus mean torque for ratchet wrench maneuver.

5.0 FUTURE DIRECTIONS

1. Extend isolated joint measurements to all joints.
2. Extend the subject pool to include a more diverse group.
3. Establish a database of dynamic isolated joint torques for general use.
4. Examine velocity dependence in more detail.

6.0 REFERENCES

1. Man-Systems Integration Standards (MSIS), NASA-STD-3000, vol. 1, 1987.
2. Malina, R.M.: Anthropometric Correlates of Strength and Motor Performance. *Exer Sport Sci Rev* vol. 3, pp. 249-274, 1975.
3. Hosler, W.W. and Morrow, J.R.: Arm and Leg Strength Compared Between Young Women and Men After Allowing for Differences in Body Size and Composition. *Ergonomics* vol. 25, pp. 309-313, 1982.
4. Pollock, M.L.; Schmidt, D.H.; and Jackson, A.S.: Measurement of Cardiorespiratory Fitness and Body Composition in the Clinical Setting. *Comprehensive Ther* vol. 6, pp.12-27, 1980.
5. LIDO Technical Report, Loredan Biomedical Inc., vol. 1 no. 1, 1987.
6. LIDO Active Operations Manual, Loredan Biomedical Inc., Sept, 1988.
7. Woolford, B.; Pandya, A.; and Maida, J.: Development of Biomechanical Models for Human Factors Evaluations. *Space Operations Applications and Research*, vol. 2, pp. 552-556, 1990.
8. Rothman, E. and Ericson, W.: *Statistics: Methods and Application*, Kendall/Hunt Publishing (Iowa), 1987.
9. Gerald, C. F.: *Applied Numerical Analysis*, Addison-Wesley Publishing Company (Menlo Park, California), 1970.
10. Pandya, A.; Hasson, S.; Aldridge, A.; Maida, J.; and Woolford, B.: The Validation of a Human Force Model to Predict Dynamic Forces Resulting from Multi-Joint Motions. NASA Technical Paper 3206.

7.0 APPENDICES

Note: All data and code presented in this paper are available in electronic form. Contact us.

APPENDIX A - Polynomial coefficients of angle versus torque collected data for one subject.

right_elbow

y

extension

4

60.000000	2.112422E+01	2.853215E-01	-1.952931E-03	0.000000E+00	0.000000E+00
120.000000	2.194190E+01	2.263627E-01	-1.558132E-03	0.000000E+00	0.000000E+00
180.000000	2.431055E+01	1.466193E-01	-1.174157E-03	0.000000E+00	0.000000E+00
240.000000	2.676228E+01	-2.846204E-02	7.660611E-05	0.000000E+00	0.000000E+00

flexion

4

60.000000	3.231453E+01	9.382667E-02	-1.072887E-03	0.000000E+00	0.000000E+00
120.000000	2.941336E+01	8.472742E-02	-1.000257E-03	0.000000E+00	0.000000E+00
180.000000	2.136176E+01	2.115664E-01	-1.517510E-03	0.000000E+00	0.000000E+00
240.000000	3.068962E+01	-1.243024E-01	1.887704E-04	0.000000E+00	0.000000E+00

right_shoulder

x

abduction

4

60.000000	6.107505E+01	-2.536474E-01	-1.324926E-03	0.000000E+00	0.000000E+00
120.000000	4.650497E+01	-1.961810E-02	-2.800043E-03	0.000000E+00	0.000000E+00
180.000000	6.524007E+01	-5.633558E-01	1.123074E-03	0.000000E+00	0.000000E+00
240.000000	5.786240E+01	-2.265438E-01	-2.315650E-03	0.000000E+00	0.000000E+00

adduction

4

60.000000	8.387678E+01	-1.146177E+00	3.783874E-03	0.000000E+00	0.000000E+00
120.000000	8.648427E+01	-8.942772E-01	3.240451E-03	0.000000E+00	0.000000E+00
180.000000	9.082626E+01	-9.451219E-01	3.533031E-03	0.000000E+00	0.000000E+00
240.000000	7.684333E+01	-8.672966E-01	3.563828E-03	0.000000E+00	0.000000E+00

flexion

4

60.000000	8.661180E+01	-6.727538E-01	1.768133E-03	0.000000E+00	0.000000E+00
120.000000	7.382412E+01	-5.487776E-01	1.419674E-03	0.000000E+00	0.000000E+00
180.000000	6.001299E+01	-3.951343E-01	1.065975E-03	0.000000E+00	0.000000E+00
240.000000	5.837907E+01	-4.092105E-01	1.305316E-03	0.000000E+00	0.000000E+00

z

lateral

4

60.000000	2.206026E+01	-1.139895E-01	-1.104627E-03	0.000000E+00	0.000000E+00
120.000000	2.191575E+01	-5.408683E-02	-1.786900E-03	0.000000E+00	0.000000E+00
180.000000	1.785649E+01	-5.820553E-02	-7.560909E-04	0.000000E+00	0.000000E+00
240.000000	1.839262E+01	-5.881445E-02	5.387657E-04	0.000000E+00	0.000000E+00

medial

4

60.000000	3.627714E+01	8.151554E-02	-4.386652E-04	0.000000E+00	0.000000E+00
120.000000	4.231037E+01	4.297858E-02	-1.839004E-03	0.000000E+00	0.000000E+00
180.000000	2.960975E+01	1.353765E-01	-1.624337E-03	0.000000E+00	0.000000E+00
240.000000	4.658150E+01	-9.520503E-02	-1.273412E-03	0.000000E+00	0.000000E+00

right_wrist

x

radial

4

60.000000	1.141442E+01	9.410274E-04	-3.824661E-03	0.000000E+00	0.000000E+00
120.000000	1.020690E+01	-5.753472E-03	-3.008693E-03	0.000000E+00	0.000000E+00
180.000000	9.970860E+00	-6.447901E-02	-1.006891E-03	0.000000E+00	0.000000E+00
240.000000	8.380389E+00	-7.270560E-02	-3.003046E-04	0.000000E+00	0.000000E+00

ulnar

4

60.000000	1.137878E+01	9.434845E-02	-2.236507E-03	0.000000E+00	0.000000E+00
120.000000	1.174232E+01	1.582629E-01	-3.338415E-03	0.000000E+00	0.000000E+00
180.000000	1.070129E+01	1.767397E-01	-2.987377E-03	0.000000E+00	0.000000E+00
240.000000	8.977362E+00	1.348160E-01	-1.860539E-03	0.000000E+00	0.000000E+00

y

extension

4

60.000000	5.737169E+00	7.455280E-02	-3.734019E-04	0.000000E+00	0.000000E+00
120.000000	5.026070E+00	5.843269E-02	-2.478556E-04	0.000000E+00	0.000000E+00
180.000000	4.333305E+00	7.833146E-02	3.712687E-04	0.000000E+00	0.000000E+00
240.000000	4.220127E+00	8.035019E-02	-1.002610E-04	0.000000E+00	0.000000E+00

flexion

4

60.000000	1.276013E+01	2.093285E-03	-1.653551E-03	0.000000E+00	0.000000E+00
120.000000	1.155679E+01	-2.462766E-02	-2.024493E-03	0.000000E+00	0.000000E+00
180.000000	1.146068E+01	-1.606610E-02	-1.736379E-03	0.000000E+00	0.000000E+00
240.000000	1.058181E+01	-5.064155E-02	-1.466442E-03	0.000000E+00	0.000000E+00

z

pronation

4

60.000000	7.311457E+00	-3.482732E-02	-6.820944E-04	0.000000E+00	0.000000E+00
120.000000	7.336191E+00	-4.664170E-02	-6.950588E-04	0.000000E+00	0.000000E+00
180.000000	6.619156E+00	-2.951976E-02	3.903088E-04	0.000000E+00	0.000000E+00
240.000000	6.058816E+00	-2.663488E-02	-3.767824E-04	0.000000E+00	0.000000E+00

supination

4

60.000000	5.309766E+00	4.141473E-02	-3.062696E-04	0.000000E+00	0.000000E+00
120.000000	5.978102E+00	3.625619E-02	-5.193476E-04	0.000000E+00	0.000000E+00
180.000000	5.724769E+00	3.905812E-02	-3.756247E-04	0.000000E+00	0.000000E+00
240.000000	6.056996E+00	4.260557E-02	-4.954465E-04	0.000000E+00	0.000000E+00

APPENDIX B - Coefficient table relating lean body mass to average torque for a particular velocity.

right_elbow					
y					
extension					
4					
60.0000	-1.780311E+01	6.688885E-01	0.000000E+00	0.000000E+00	
120.0000	-1.777659E+01	6.585007E-01	0.000000E+00	0.000000E+00	
180.0000	-1.575342E+01	6.043151E-01	0.000000E+00	0.000000E+00	
240.0000	-1.138565E+01	5.154829E-01	0.000000E+00	0.000000E+00	
flexion					
4					
60.0000	-1.170113E+01	6.214280E-01	0.000000E+00	0.000000E+00	
120.0000	-1.499996E+01	6.524455E-01	0.000000E+00	0.000000E+00	
180.0000	-1.190419E+01	5.645015E-01	0.000000E+00	0.000000E+00	
240.0000	-9.734751E+00	4.945650E-01	0.000000E+00	0.000000E+00	
right_shoulder					
x					
abduction					
4					
60.0000	-1.162713E+01	6.525019E-01	0.000000E+00	0.000000E+00	
120.0000	-1.005819E+01	5.898796E-01	0.000000E+00	0.000000E+00	
180.0000	-9.859225E+00	5.921002E-01	0.000000E+00	0.000000E+00	
240.0000	-8.557056E+00	5.585406E-01	0.000000E+00	0.000000E+00	
adduction					
4					
60.0000	-1.840421E+01	8.565667E-01	0.000000E+00	0.000000E+00	
120.0000	-1.617023E+01	8.101834E-01	0.000000E+00	0.000000E+00	
180.0000	-2.081728E+01	8.792474E-01	0.000000E+00	0.000000E+00	
240.0000	-1.698798E+01	8.091762E-01	0.000000E+00	0.000000E+00	
y					
extension					
4					
60.0000	-2.438441E+01	9.667410E-01	0.000000E+00	0.000000E+00	
120.0000	-2.267868E+01	9.266518E-01	0.000000E+00	0.000000E+00	
180.0000	-1.129235E+01	6.457043E-01	0.000000E+00	0.000000E+00	
240.0000	-2.071749E+01	8.310267E-01	0.000000E+00	0.000000E+00	
flexion					
4					
60.0000	-1.827386E+01	8.346249E-01	0.000000E+00	0.000000E+00	
120.0000	-2.118529E+01	8.891425E-01	0.000000E+00	0.000000E+00	
180.0000	-9.366316E+00	6.267306E-01	0.000000E+00	0.000000E+00	
240.0000	-1.645339E+01	7.376269E-01	0.000000E+00	0.000000E+00	
z					
lateral					
4					
60.0000	-8.936315E+00	4.264861E-01	0.000000E+00	0.000000E+00	
120.0000	-7.174304E+00	3.740323E-01	0.000000E+00	0.000000E+00	
180.0000	-4.212536E+00	3.108540E-01	0.000000E+00	0.000000E+00	
240.0000	-6.357865E+00	3.512995E-01	0.000000E+00	0.000000E+00	
medial					
4					
60.0000	-1.540060E+01	6.548921E-01	0.000000E+00	0.000000E+00	
120.0000	-1.114382E+01	5.727975E-01	0.000000E+00	0.000000E+00	
180.0000	-1.053552E+01	5.327551E-01	0.000000E+00	0.000000E+00	
240.0000	-1.421859E+01	6.126382E-01	0.000000E+00	0.000000E+00	
right_wrist					
x					

radial

4

60.0000	-1.268475E+00	1.357690E-01	0.000000E+00	0.000000E+00
120.0000	-1.005637E+00	1.208873E-01	0.000000E+00	0.000000E+00
180.0000	-1.270290E+00	1.214390E-01	0.000000E+00	0.000000E+00
240.0000	-1.770912E+00	1.263918E-01	0.000000E+00	0.000000E+00

ulnar

4

60.0000	-2.900480E+00	1.932900E-01	0.000000E+00	0.000000E+00
120.0000	-3.978939E+00	2.042846E-01	0.000000E+00	0.000000E+00
180.0000	-2.994360E+00	1.823847E-01	0.000000E+00	0.000000E+00
240.0000	-3.975077E+00	1.952298E-01	0.000000E+00	0.000000E+00

y

extension

4

60.0000	-1.127101E+00	8.416442E-02	0.000000E+00	0.000000E+00
120.0000	-7.367646E-01	7.451715E-02	0.000000E+00	0.000000E+00
180.0000	-1.423397E+00	8.594213E-02	0.000000E+00	0.000000E+00
240.0000	-1.275451E+00	8.026770E-02	0.000000E+00	0.000000E+00

flexion

4

60.0000	4.747616E+00	2.070452E-01	0.000000E+00	0.000000E+00
120.0000	-4.260902E+00	1.903496E-01	0.000000E+00	0.000000E+00
180.0000	-2.040817E+00	1.303384E-01	0.000000E+00	0.000000E+00
240.0000	-3.654124E+00	1.715256E-01	0.000000E+00	0.000000E+00

z

pronation

4

60.0000	-6.592008E-01	7.739327E-02	0.000000E+00	0.000000E+00
120.0000	-1.366900E+00	9.181561E-02	0.000000E+00	0.000000E+00
180.0000	-1.853492E+00	1.013228E-01	0.000000E+00	0.000000E+00
240.0000	-2.182193E+00	1.070073E-01	0.000000E+00	0.000000E+00

supination

4

60.0000	-2.369235E+00	1.065645E-01	0.000000E+00	0.000000E+00
120.0000	-2.446581E+00	1.088279E-01	0.000000E+00	0.000000E+00
180.0000	-2.280463E+00	1.070542E-01	0.000000E+00	0.000000E+00
240.0000	-2.641539E+00	1.119357E-01	0.000000E+00	0.000000E+00

Appendix C - Normalized torque coefficients for entire subject pool.

right_elbow

y

extension

4

60.0000	2.382654E-01	1.609246E-02	-7.057701E-05	0.000000E+00	0.000000E+00
120.0000	3.998269E-01	1.352749E-02	-6.401985E-05	0.000000E+00	0.000000E+00
180.0000	6.186674E-01	7.465366E-03	-2.948732E-05	0.000000E+00	0.000000E+00
240.0000	6.797335E-01	5.123406E-03	-1.389026E-05	0.000000E+00	0.000000E+00

flexion

4

60.0000	9.436899E-01	1.385255E-03	-7.300826E-06	0.000000E+00	0.000000E+00
120.0000	9.559348E-01	2.457487E-03	-1.990186E-05	0.000000E+00	0.000000E+00
180.0000	1.024970E+00	7.605217E-04	-1.120966E-05	0.000000E+00	0.000000E+00
240.0000	1.181190E+00	-2.677844E-03	3.123384E-06	0.000000E+00	0.000000E+00

right_shoulder

x

abduction

4

60.0000	1.638217E+00	-9.234213E-03	1.584508E-05	0.000000E+00	0.000000E+00
120.0000	1.548579E+00	-6.328132E-03	-3.656676E-06	0.000000E+00	0.000000E+00
180.0000	1.754382E+00	-1.159083E-02	2.188312E-05	0.000000E+00	0.000000E+00
240.0000	1.667307E+00	-9.909423E-03	1.595937E-05	0.000000E+00	0.000000E+00

adduction

4

60.0000	8.051392E-01	4.159272E-03	-1.923175E-05	0.000000E+00	0.000000E+00
120.0000	8.114389E-01	4.182187E-03	-2.022592E-05	0.000000E+00	0.000000E+00
180.0000	8.734385E-01	1.717887E-03	-2.843898E-06	0.000000E+00	0.000000E+00
240.0000	7.575982E-01	4.319388E-03	-1.588424E-05	0.000000E+00	0.000000E+00

y

extension

4

60.0000	6.296859E-01	7.588339E-03	-3.253375E-05	0.000000E+00	0.000000E+00
120.0000	4.782509E-01	9.294314E-03	-3.527058E-05	0.000000E+00	0.000000E+00
180.0000	7.084687E-01	4.127454E-03	-1.147116E-05	0.000000E+00	0.000000E+00
240.0000	6.559435E-01	3.129994E-03	-1.157555E-06	0.000000E+00	0.000000E+00

flexion

4

60.0000	1.873006E+00	-1.086977E-02	2.096022E-05	0.000000E+00	0.000000E+00
120.0000	1.857643E+00	-1.102275E-02	2.299863E-05	0.000000E+00	0.000000E+00
180.0000	1.801719E+00	-1.150634E-02	2.964644E-05	0.000000E+00	0.000000E+00
240.0000	1.710937E+00	-9.788573E-03	2.236658E-05	0.000000E+00	0.000000E+00

z

lateral

4

60.0000	1.061949E+00	-5.109649E-03	-1.644733E-05	0.000000E+00	0.000000E+00
120.0000	1.041042E+00	-4.551276E-03	-1.793456E-05	0.000000E+00	0.000000E+00
180.0000	1.007360E+00	-4.989951E-03	9.017851E-06	0.000000E+00	0.000000E+00
240.0000	9.669892E-01	-4.986339E-03	2.867537E-05	0.000000E+00	0.000000E+00

medial

4

60.0000	1.022044E+00	4.982772E-03	-3.170801E-05	0.000000E+00	0.000000E+00
120.0000	1.041949E+00	4.882844E-03	-5.522752E-05	0.000000E+00	0.000000E+00
180.0000	1.029437E+00	4.798532E-03	-4.722543E-05	0.000000E+00	0.000000E+00
240.0000	1.021175E+00	3.547545E-03	-3.615252E-05	0.000000E+00	0.000000E+00

right_wrist

x

radial

4					
60.0000	1.181802E+00	-1.675913E-02	-6.016592E-04	0.000000E+00	0.000000E+00
120.0000	1.107246E+00	-1.877522E-02	-4.386299E-04	0.000000E+00	0.000000E+00
180.0000	1.036549E+00	-1.984581E-02	-3.919646E-04	0.000000E+00	0.000000E+00
240.0000	1.039976E+00	-2.084383E-02	-3.424259E-04	0.000000E+00	0.000000E+00
ulnar					
4					
60.0000	1.066880E+00	9.789578E-03	-3.174051E-04	0.000000E+00	0.000000E+00
120.0000	1.087365E+00	1.114404E-02	-3.558657E-04	0.000000E+00	0.000000E+00
180.0000	1.114945E+00	1.416128E-02	-3.476919E-04	0.000000E+00	0.000000E+00
240.0000	1.073311E+00	1.608139E-02	-3.904515E-04	0.000000E+00	0.000000E+00
y					
extension					
4					
60.0000	9.806204E-01	8.960170E-03	-5.493223E-05	0.000000E+00	0.000000E+00
120.0000	9.499655E-01	9.603754E-03	-4.643842E-05	0.000000E+00	0.000000E+00
180.0000	9.177343E-01	8.428668E-03	-3.641872E-06	0.000000E+00	0.000000E+00
240.0000	9.216082E-01	9.430406E-03	-2.801543E-05	0.000000E+00	0.000000E+00
flexion					
4					
60.0000	1.084562E+00	4.124617E-04	-1.078120E-04	0.000000E+00	0.000000E+00
120.0000	1.109554E+00	5.107403E-04	-1.394872E-04	0.000000E+00	0.000000E+00
180.0000	1.091251E+00	-5.016060E-05	-1.093535E-04	0.000000E+00	0.000000E+00
240.0000	1.088930E+00	-9.377312E-04	-1.080113E-04	0.000000E+00	0.000000E+00
z					
pronation					
4					
60.0000	1.010391E+00	5.380042E-03	3.692308E-06	0.000000E+00	0.000000E+00
120.0000	9.950561E-01	5.733134E-03	7.248831E-06	0.000000E+00	0.000000E+00
180.0000	1.009005E+00	5.072503E-03	-6.918676E-06	0.000000E+00	0.000000E+00
240.0000	9.778854E-01	6.124177E-03	1.015605E-05	0.000000E+00	0.000000E+00
supination					
4					
60.0000	1.024255E+00	-7.277947E-03	-3.147846E-05	0.000000E+00	0.000000E+00
120.0000	9.929054E-01	-7.556385E-03	-3.043222E-05	0.000000E+00	0.000000E+00
180.0000	1.009055E+00	-7.271954E-03	-5.403312E-05	0.000000E+00	0.000000E+00
240.0000	9.361496E-01	-6.824119E-03	-1.593118E-05	0.000000E+00	0.000000E+00

Appendix D - Source code for utilization of torque coefficient tables. This code is available compiled on UNIX, IBM PC, and Macintosh computers. Contact us.

```
#include <stdio.h>
#include <math.h>
#include "lean.h"
```

```
/******
```

Authors Abhilash Pandya, James Maida.

Date: 8/29/91

Note: You are welcome to copy and modify this set of routines to fit your own needs.

This is a set of tool kit programs to allow easy manipulation of the lean body mass to torque relationship (see report). It creates coefficient files which describe fully each joint. The coefficient files relate torque for a particular joint to its angle and velocity. See the discussion on ffc files and the header file which describes the ffc files.

The main program is an example of the way the tool kit can be utilized.

It does the following operations:

- 1 -> Input a new %bf and weight
- 2 -> Print Population Coefficients
- 3 -> Print Lean Body Mass coefficients
- 4 -> Print This individual's coefficients
- 5 -> Write out (*.ffc files) individuals coefficients
- 6 -> Print out a torque summary for this individual

Units: weight- kg. torque - ft-lbs.

These routines include only the standard C files and should compile on most machines in its present form.

to compile use :

```
cc lean.c leanlib.c -lm -o lean
```

to execute use : lean <%body fat> < weight >

For Mac: Think C compiler, make sure to include the ANSI Library in your project file and make sure that the ANSI library is split into its own code segment (There is a 32 K code segment limit. see you manuals for details).

```
*****
```

```
/*add to this list as more joints become available and increment JC the joint count*/
```

```
static char joint_name[MAXJOINT][80] = {"right_shoulder",
                                         "right_elbow",
                                         "right_wrist"};
```

```
static AxisTable AxisMap[] =
```

```
{
  "ABD VEL",      "abduction", "x", 1,
  "ADD VEL",      "adduction", "x", -1,
  "FLEX VEL",     "flexion",   "y", 1,
  "EXT VEL",      "extension", "y", -1,
  "EXT ROT",      "lateral",   "z", 1,
  "INT ROT",      "medial",    "z", -1,
  "PRONATION VELOCITY", "pronation", "z", 1,
  "SUPINATION VELOCITY", "supination", "z", -1,
  "RADIAL DEV",   "radial",    "x", 1,
  "ULNAR DEV",    "ulnar",     "x", -1,
  0,              0,          0, 0
};
```

```
int AxisTableSize = 10;
```

CalculateIndividualCoeffTable (itable, ptable, ltable, bf, weight, count)

```
CoeffTable itable[];  
CoeffTable ptable[];  
CoeffTable ltable[];  
float bf;  
float weight;  
int count;
```

Function:

Given the population normalized table (ptable) and the lean body mass to torque relationship table (ltable), this program will calculate the the itable (individual person's table) from his/her body fat and weight.

The itable is calculated by computing the average torque for each joint direction and velocity from the ltable. The ltable (lean body mass table) contains coefficients which describe the relationship between the average torque and the lean body mass. To use this table, first a lean body mass is computed from the weight and % body fat. This value is plugged into the coefficients looked up in the ltable. This yields the average torque for that joint direction and velocity. After this torque is computed, the normalized torque coefficients for that joint, direction and velocity are looked up in the ptable and multiplied by the average torque computed from the ltable. These coefficients are then stored into the itable and returned.

Parameters:

itable - individuals table
ptable - population/ normalized table
ltable - lean body mass to average torque table
bf - body fat.
mass - mass of the person.

```
{  
    int axis;  
    int dir;  
    int eq_count;  
    int i, j, z;  
    float lean_body_mass, average;  
    float ComputeCoeff ()  
  
    /*lean body mass calculation*/  
    lean_body_mass = weight * ( 1.0 - bf );  
    printf ( "Coefficients calculated for lean body mass %f", lean_body_mass);  
    /*for each joint*/  
    for (j = 0 ; j < count; j++)  
    {  
        /*copy the name and axis count to the itable*/  
        strcpy ( itable[j].joint_name, ptable[j].joint_name);  
        itable[j].axis_count = ptable[j].axis_count;  
        /*for each axis in this joint*/  
        for( axis = 0; axis < ptable[j].axis_count; axis++ )  
        {  
            /*copy the axis number*/  
            itable[j].axis[axis] = ptable[j].axis[axis];  
        }  
    }  
}
```

```

/*for each direction in this axis*/
for( dir = 0; dir < 2; dir++ )
{
    /*copy the direction name and equation count to the ptable*/
    strcpy ( itable[j].function[axis][dir].direction_name,
            ptable[j].function[axis][dir].direction_name);
    itable[j].function[axis][dir].eq_count =
        ptable[j].function[axis][dir].eq_count;
    eq_count = ptable[j].function[axis][dir].eq_count;

    /*for each regression equation compute the individuals eqns*/

    for( i = 0; i < eq_count; i++ )
    {
        /*copy the velocity information*/
        itable[j].function[axis][dir].velocity[i] =
            ptable[j].function[axis][dir].velocity[i];
        /*compute the average for torque for this dir, velocity*/
        /*from the lean body mass information and the ltable */
        average =
            ComputeCoeff (itable[j].function[axis][dir].coeff[i],
                          lean_body_mass);
        /*multiply the coefficients of the pop table by average*/
        /*and store in the itable */
        MultiplyCoeff (itable[j].function[axis][dir].coeff[i],
                      ptable[j].function[axis][dir].coeff[i],
                      average);
    } /*end for i*/

} /*end for dir*/

} /* end for axis*/

} /*end for j-joint*/
}

```

```

WriteCoeffTableToFile( table, ext, count )
CoeffTable table[];
char ext[];
int count;

```

Function:

Given a table pointer a file extension, and the number of joints, it will write the coefficients of the table specified into files with the names of the files taken from the joint names in joint_name table with the extension in variable ext.

This is an example of the file format:

right_elbow

Y

extension

4

60.000000 1.743286e+01 1.177417e+00 -5.163818e-03 0.000000e+00

120.000000 2.869938e+01 9.709965e-01 -4.595313e-03 0.000000e+00

180.000000 4.110020e+01 4.959499e-01 -1.958944e-03 0.000000e+00

240.000000 3.991397e+01 3.008465e-01 -8.156364e-04 0.000000e+00

extension

4

```
60.000000 1.743286e+01 1.177417e+00 -5.163818e-03 0.000000e+00
120.000000 2.869938e+01 9.709965e-01 -4.595313e-03 0.000000e+00
180.000000 4.110020e+01 4.959499e-01 -1.958944e-03 0.000000e+00
240.000000 3.991397e+01 3.008465e-01 -8.156364e-04 0.000000e+00
```

line 1: joint name

line 2: axis

line 3: direction

line 4: number of regression equations.

line 5: velocity, coefficient 1 coefficient 2 ...

this repeats until the entire joint is described.

Parameters:

table - storage table for the coefficient values.

ext - file extension the data to be stored in (eg ffc , ntc or lbc)

count - number of joints.

```
{
    FILE *fp;
    char filename[80];
    char line[132];
    int i, j;
    int axis;
    char axisname[32];
    char dirname[80];
    int ano;
    int dir;
    int eq_count;
    int joint;
    /*for each joint in the joint names,
    write the file for that joint
    for the table specified.*/
    for ( joint = 0 ; joint < count ; joint++)
    {
        /*open a file for this joint using names form the
        joint_name table
        */
        strcpy ( filename, joint_name[joint]);
        sprintf( filename, "%s.%s", filename, ext );
        fp = fopen( filename, "w" );
        if( fp == NULL )
        {
            printf( "failed open on %s\n", filename );
            return( 0 );
        }
        /* write out name of joint */
        fprintf( fp, "%s\n", table[joint].joint_name );
        for (axis = 0 ; axis < table[joint].axis_count ; axis++ )

            if ( table[joint].axis[axis]== 0 ) fprintf(fp, "X\n");
        if ( table[joint].axis[axis]== 1 ) fprintf(fp, "Y\n");
        if ( table[joint].axis[axis]== 2 ) fprintf(fp, "Z\n");

        /* direction name */
        for( dir = 0; dir < 2; dir++ )
        {
```

```

        fprintf(fp, "%s\n", table[joint].function[axis][dir].direction_name );

/* number of force equations for this direction */

        fprintf( fp, "%d\n",
            table[joint].function[axis][dir].eq_count );
        eq_count = table[joint].function[axis][dir].eq_count;

        for( i = 0; i < eq_count; i++ )
        {
            fprintf( fp, "%f %e %e %e %e \n",
                table[joint].function[axis][dir].velocity[i],
                table[joint].function[axis][dir].coeff[i][0],
                table[joint].function[axis][dir].coeff[i][1],
                table[joint].function[axis][dir].coeff[i][2],
                table[joint].function[axis][dir].coeff[i][3] );
        }
    } /* axis loop */
fclose (fp);
} /*end for each joint*/
}

TorqueSummary ( table, bf, weight, count)
CoeffTable table[];
float bf;
float weight;
int count;

/*****
Function: Print out a quick-look table of the average torque values
for each of the joints in the joint chain. This function takes the lean
body mass table coefficients the %bf, and weight of an individual and computes
the average torque for joint, direction. All velocities are averaged before
being printed. This means that the dynamic information is not printed in order
to keep it a quick look table.
Parameters:
table - the lean body mass table.
bf - % body fat ( 0 - 1)
weight - weight of the person.
count - joint count.
*****/

{
int axis;
int dir;
int eq_count;
int i, j;
float average, avg;
float ComputeCoeff ();
float lean_body_mass;
lean_body_mass = weight * ( 1.0 - bf );
printf ("\n\n");
printf ("-----\n");
printf (" Weight %f Body Fat %f Lean Mass %f\n", weight, bf, lean_body_mass);
for (j = 0 ; j < count; j++)
{
    printf( "\n\nJoint Name: %s\n", table[j].joint_name );

```

```

for( axis = 0; axis < table[j].axis_count; axis++ )
{
    for( dir = 0; dir < 2; dir++ )
    {
        printf( "Direction: %s",
            table[j].function[axis][dir].direction_name );

        eq_count = table[j].function[axis][dir].eq_count;
        for( i = 0; i < eq_count; i++ )
        {
            /*compute the average for torque for this dir, velocity*/
            average =
                ComputeCoeff (table[j].function[axis][dir].coeff[i],
                    lean_body_mass);
            avg +=average;
        }
        printf ( " Average Torque: %f\n", avg / eq_count);
        avg = 0.0;
    }
}
}
}

printf ("-----\n");
}

```

ReadCoeffTable(table, ext, count)

/*****

Function:

Given a table pointer a file extension, and the number of joints, it will fill the coefficient table specified with the values in the file according to the joint names in the joint_name table defined.

Parameters:

table - storage table for the coefficient values.

ext - file extension the data are stored in (eg ffc , ntc or lbc)

count - number of joints.

/*****

CoeffTable table[];

char ext[];

int count;

{

FILE *fp;

char filename[80];

char line[132];

int i, j;

char *ler;

int axis;

char axisname[32];

char dirname[80];

int ano;

int dir;

int eq_count;

int joint;

/* initialize force table */

```

InitJointCoeffTable( table );
/*for each joint in the joint names,
load the file for that joint
into the table specified.*/

for ( joint = 0 ; joint < count ; joint++)
{
    strcpy ( filename, joint_name[joint]);
    sprintf( filename, "%s.%s", filename, ext );
    fp = fopen( filename, "r" );
    if( fp == NULL )
    {
        printf( "failed open on %s\n", filename );
        return( 0 );
    }
    /* read in name of joint */
    ier = fgets( line, 132, fp );
    if( ier == NULL ) return( 0 );
    sscanf( line, "%s\n", table[joint].joint_name );
    /* read remaining data in file */
    axis = 0;
    while( 1 )
    {
        ier = fgets( line, 132, fp );
        if( ier == NULL ) break;
        /* axis number */
        sscanf( line, "%s", axisname );
        ano = FTGetAxisNumber( axisname[0] );
        if( ano == -1 )
        {
            printf( "invalid axis name %s\n", axisname );
            printf( "using default X axis\n" );
            ano = 0;
        }
        table[joint].axis[axis] = ano;
        for( j = 0; j < 2; j++ )
        {
            ier = fgets( line, 132, fp );
            if( ier == NULL ) break;
            /* direction name */
            sscanf( line, "%s", dirname );
            /* get direction code 0 = negative, 1 = positive, 2 = error */
            dir = GetAxisDirection( dirname );
            if( dir == 2 ) dir = 0;
            strcpy( table[joint].function[axis][dir].direction_name,
                    dirname );
            ier = fgets( line, 132, fp );
            if( ier == NULL ) break;

            /* number of force equations for this direction */
            sscanf( line, "%d",
                    &table[joint].function[axis][dir].eq_count );
            eq_count = table[joint].function[axis][dir].eq_count;
            for( i = 0; i < eq_count; i++ )
            {
                ier = fgets( line, 132, fp );
                if( ier == NULL ) break; /* break out of for loop */
            }
        }
    }
}

```

```

                                sscanf( line, "%f %e %e %e %e",
                                    &table[joint].function[axis][dir].velocity[i],
                                    &table[joint].function[axis][dir].coeff[i][0],
                                    &table[joint].function[axis][dir].coeff[i][1],
                                    &table[joint].function[axis][dir].coeff[i][2],
                                    &table[joint].function[axis][dir].coeff[i][3] );
                                }
                                }
                                /* break out of the while loop */
                                if( ier == NULL ) break;
                                axis++;
                                table[joint].axis_count = axis;
                                }
                                fclose (fp);

                                } /*end for each joint*/
}

int InitJointCoeffTable ( table )
/*****
Function: Given a pointer to the coefficient table, this function
will put all the appropriate initial values into the table.
Parameters:
table - pointer to the coefficient table to be initialized.
*****/
CoeffTable table[];
{
    int i, j, k, l, m ;
    /*for each joint in the table*/
    for (m = 0; m < MAXJOINT; m++)
    {
        table[m].joint_name[0] = 0;
        table[m].axis_count = 0;
        table[m].axis[0] = 0;
        table[m].axis[1] = 1;
        table[m].axis[2] = 2;
        /*for each axis in that joint*/
        for( l = 0; l < AXIS; l++ )
        {
            /*for each direction in that axis*/
            for( j = 0; j < DIR; j++ )
            {
                table[m].function[i][j].eq_count = 0;
                table[m].function[i][j].direction_name[0] = 0;
                /*for each velocity in that direction*/
                for( k = 0; k < VELOCITY; k++ )
                {
                    table[m].function[i][j].velocity[k] = 0.0;
                    for( l = 0; l < COEFF; l++ )
                        table[m].function[i][j].coeff[k][l] = 0.0;
                }
            }
        }
    }
}

```

```

ListCoeffTable( table, label, count )
CoeffTable table[];
char label[];
int count;
/*****
Function: Given a pointer to a table, this function
writes the function to stdout.
Parameters:
table - pointer to a table.
*****/
{
    int axis;
    int dir;
    int eq_count;
    int i, j;
    printf ( "%s\n", label);
    for (j = 0 ; j < count; j++)
    {
        printf( "Joint Name: %s\n", table[j].joint_name );
        printf( "DOF's : %d\n", table[j].axis_count );
        for( axis = 0; axis < table[j].axis_count; axis++)
        {
            for( dir = 0; dir < 2; dir++ )
            {
                printf( " Direction: %s\n",
                    table[j].function[axis][dir].direction_name );
                printf( " Number of Functions: %d\n",
                    table[j].function[axis][dir].eq_count );
                printf( " Velocity ----- Coefficients ----- \n" );
                eq_count = table[j].function[axis][dir].eq_count;
                for( i = 0; i < eq_count; i++ )
                {
                    printf( " %f %E %E %E %E\n",
                        table[j].function[axis][dir].velocity[i],
                        table[j].function[axis][dir].coeff[i][0],
                        table[j].function[axis][dir].coeff[i][1],
                        table[j].function[axis][dir].coeff[i][2],
                        table[j].function[axis][dir].coeff[i][3] );
                }
            }
        }
    }
}

float ipower( x, y )
float x;
int y;
/*****
float ipower( x, y ) - compute x to the y power
Function:
Compute the value of x raised to the y'th power.
*****/
{
    float z;
    z = 1.0;
    while( y-- ) { z *= x; }
    return( z );
}

```

```

MultiplyCoeff ( coeff1 , coeff2, value)
float coeff1[];
float coeff2[];
float value;
/*****
Function:
Multiply the array of coeffs by the value specified.
coeff1 = coeff2 * value;
Parameters:
coeff - an array of float values that are the coefficients to
an n degree polynomial.
value - the value to multiply with.
*****/
{
int i;
for( i = 0; i < COEFF; i++ )
{
coeff1[i] =coeff2[i] * value;
}
}

float ComputeCoeff ( coeff, value )
float coeff[];
float value;

/*****
Function:
Solve the coefficients for the value specified.
Parameters:
coeff - an array of float values that are the coefficients to
an n degree polynomial.
value - the x value of the function to compute the y.
*****/
{
int i;
float answer;
answer = coeff[0];
for( i = 1; i < COEFF; i++ )
{
answer += ipower( value , i ) * coeff[i];
}
return ( answer );
}

int FTGetAxisNumber( name )
char name;
/*****
int FTGetAxisNumber( name ) Function:
Given an axis name return axis number associated with name.
*****/
{
char a;

a = toupper( name );
if ( a == 'X' ) return( 0 );
if ( a == 'Y' ) return( 1 );
if ( a == 'Z' ) return( 2 );
return( -1 );
}

```

```

}

int GetAxisDirection( name )
*****
Function: Given the name of the direction, return a direction number.
Parameters: name - direction name.
*****
char *name;
{
    int i;
    int k;
    i = 0;
    while( AxisMap[i].input_name[0] )
    {
        k = strlen( AxisMap[i].output_name );
        if( strcmp( name, AxisMap[i].output_name, k ) == 0 )
        {
            if( AxisMap[i].direction < 0 ) return( AxisMap[i].direction + 1 );
            else return( AxisMap[i].direction );
        }
        i++;
    }
    return( 2 );
}

/*****
GetVelocityCode( joint, axis, dir, velocity )
Function:
    Get the code number associated with a given velocity
    found in the force function velocity table. The code
    number is selected by comparing the given velocity with
    the velocities in the table and choosing the index number
    of the closest value.
*****/
int GetVelocityCode( force_table, joint, axis, dir, velocity )
CoeffTable force_table[];
int joint;
int axis;
int dir;
float velocity;
{
    int i;
    int count;
    float mid;
    /*
    size of the velocity is the same as the number of
    equations within the given axis and direction
    */
    count = force_table[joint].function[axis][dir].eq_count;
    /* check for special case of zero vel. */
    /*no equations*/
    if( count == 0 ) return ( -1 );
    if( velocity < force_table[joint].function[axis][dir].velocity[0] )
        return( 0 );
    /* scan table */
    for( i = 0; i < count - 1; i++ )
    {
        if( (velocity > force_table[joint].function[axis][dir].velocity[i]) &&

```

```

        (velocity < force_table[joint].function[axis][dir].velocity[i+1]) )
    {
/* mid = (v[i+1] - v[i]) / 2.0 + v[i] */
        mid = ( force_table[joint].function[axis][dir].velocity[i+1] -
                force_table[joint].function[axis][dir].velocity[i]) / 2.0 +
                force_table[joint].function[axis][dir].velocity[i];
/* velocity is on one side or the other of mid point */
        if( velocity < mid ) return( i );
        return( i + 1 );
    }
}
return( count - 1 ); /* need some default */
}
/*****
int GetCoeff( joint, axis, dir, velocity, coeff )
Function:
    return the force coefficents associated with the given
    joint, axis, direction and velocity.
/*****
int GetCoeff(force_table, joint, axis, dir, velocity, coeff )
CoeffTable force_table[];
int joint;
int axis;
int dir;
float velocity;
float coeff[];
{
    int vid;
    int i;
    int GetVelocityCode();
    vid = GetVelocityCode( joint, axis, dir, velocity );
    for( i = 0; i < COEFF; i++ )
    {
        coeff[i] = force_table[joint].function[axis][dir].coeff[vid][i];
    }
    return( vid );
}

/*****MAIN*****/
#include <stdio.h>
#include <math.h>
#include "lean.h"
menu ()
/*****
Function: Print a menu.
Parameters: none.
*****/
{
    printf ( " \n\n");
    printf ( "Fat Menu =====> :-) \n");
    printf ( "0 -> exit. \n");
    printf ( "1 -> Input a new bf and weight\n\n");
    printf ( "2 -> Print Population Coefficients\n");
    printf ( "3 -> Print Lean Body Mass coefficients\n");
    printf ( "4 -> Print This individuals coefficients\n\n");
    printf ( "5 -> Write out (*.ffc files) individuals coefficients\n");
    printf ( "6 -> Print out a torque summary for this individual.\n");
}

```

```

}
main (argc, argv)
int argc;
char *argv[];
/*****
Function: main program. Menu selection and execution.
Parameters: body fat and weight.
*****/

{
/*****
type definitions for CoeffTable data structure see lean.h.
Decription of force table data structure.
table[joint].function[axis][dir]
    force functions associated with a joint, these
    are indexed by the axis number and the direction
    code.
table[joint].joint_name
    name of joint 80 characters max.
table[joint].axis[axis]
    table of axis numbers (DOF's) for the joint.
table[joint].axis_count
    number of axes or DOF's for the joint.
function[axis][dir].coeff[vid][coeff]
    force function coeffcients index by the velocity code (vid)
    (see velocity code) and coefficient index (1-4).
function[axis][dir].velocity[vid]
    velocity value associated with a given velocity code (vid).
function[axis][dir].eq_count
    number of force function equations associated with
    a given axis and direction.
function[axis][dir].direction_name
    name of a given direction, i.e. extension, flexion, etc.
*****/

CoeffTable pop_table[MAXJOINT]; /*normalized population coefficients*/
CoeffTable lean_table[MAXJOINT]; /*relationship: mean torque to lean mass*/
CoeffTable ind_table[MAXJOINT]; /*Calculated coeff table based on
    weight and %body fat*/
float bf; /* body fat percentage*/
float weight; /*weight*/
int done; /*loop control */
int option; /*user input */
#if (PC || UNIX)
/*process command line arguments*/
if ( argc < 3 )
{
    printf ( "usage: lean <%body_fat> <weight> \n");
    exit( 0 );
}
#endif
#if MAC
printf ( "\n\n");
printf ("input body fat (0-1) and <weight(kg):");
scanf ( "%f %f", &bf, &weight);
printf ( "%f %f", bf, weight);
#else
bf = atof ( argv[1] );
weight =(float) atof ( argv[2]);

```

```

#endif
if (bf > 1)
{
    printf ( " invalid % body fat :-) enter value between 0-1 \n");
    exit( 0 );
}
/*read the two tables 1. Population table 2. lean body table*/
ReadCoeffTable ( pop_table , "ntc" , JC );
ReadCoeffTable ( lean_table, "lbc", JC );
/*calculate this individual's table*/
InitJointCoeffTable ( ind_table );
CalculateIndividualCoeffTable
(ind_table, pop_table, lean_table,
bf, weight, JC );
done = FALSE;
while ( !done )
{
    menu ();
    scanf ( "%d", &option);
    switch (option)
    {
        case 0: /*exit*/
            printf ("May the \"lean force\" be with you. \n");
            done = TRUE;
            break;
        case 1: /*Input new values*/
            printf ("Input body fat (0 - 1) and weight:");
            scanf ("%f %f", &bf, &weight);
            CalculateIndividualCoeffTable
            (ind_table, pop_table, lean_table,
            bf, weight, JC );
            break;
        case 2: /*Print Population Coefficients*/
            ListCoeffTable ( pop_table, "Population Table", JC);
            break;
        case 3: /*Print Lean Body mass coefficients */
            ListCoeffTable ( lean_table, "Lean Body mass -> average torque", JC);
            break;;
        case 4: /*Print Individuals coefficients*/
            ListCoeffTable ( ind_table, "Individuals coefficients", JC);
            break;
        case 5: /*Write out individuals coefficient file*/
            WriteCoeffTableToFile( ind_table, "ffc", JC);
            printf ( " *.ffc files are written\n");
            break;
        case 6: /*Write out a summary table for this individual*/
            TorqueSummary ( lean_table, bf, weight, JC);
            break;
        default:
            printf (" TRY AGAIN. (enter options 0 ->6) \n");
            break;
    }
}
}
}

```